The background of the cover is a photograph of the Golden Gate Bridge in San Francisco. The bridge's iconic orange-red towers and suspension cables are prominent, extending from the left side of the frame towards the right. In the distance, the San Francisco city skyline is visible, including the Transamerica Pyramid. The water of the bay is a deep blue, and the sky is clear and light blue. The overall scene is captured from a high angle, looking down at the bridge and across the bay.

SEKE

San Francisco Bay
July 1-3 2016

Proceedings of the Twenty-Eighth
International Conference on
Software Engineering &
Knowledge Engineering

PROCEEDINGS
SEKE 2016

**The 28th International Conference on
Software Engineering &
Knowledge Engineering**

Sponsored by

KSI Research Inc. and Knowledge Systems Institute Graduate School, USA

Technical Program

July 1 – 3, 2016

Hotel Sofitel, Redwood City, San Francisco Bay, USA

Organized by

KSI Research Inc. and Knowledge Systems Institute Graduate School, USA

Copyright © 2016 by KSI Research Inc. and Knowledge Systems Institute Graduate School

All rights reserved. No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, electronic, mechanical, photocopying, recording, or otherwise, without the prior written consent of the publisher.

ISBN: 1-891706-39-X

ISSN: 2325-9000 (print)

2325-9086 (online)

DOI reference number: 10.18293/SEKE2016

Publisher Information:

KSI Research Inc. and Knowledge Systems Institute Graduate School

156 Park Square

Pittsburgh, PA 15238 USA

Tel: +1-412-606-5022

Fax: +1-847-679-3166

Email: seke@ksiresearch.org

Web: <http://ksiresearchorg.ipage.com/seke/seke16.html>

Proceedings preparation, editing and printing are sponsored by KSI Research Inc. and Knowledge Systems Institute Graduate School, USA.

Printed by KSI Research Inc. and Knowledge Systems Institute Graduate School

FOREWORD

Welcome to the 28th International Conference on Software Engineering and Knowledge Engineering (SEKE), in Hotel Sofitel, Redwood City, USA. In last 28 years, SEKE has established itself as a major international forum to foster, among academia, industry, and government agencies, discussion and exchange of ideas, research results and experience in software engineering and knowledge engineering. The SEKE community has grown to become a very important and influential source of ideas and innovations on the interplays between software engineering and knowledge engineering, and its impact on the knowledge economy has been felt worldwide. On behalf of the Program Committee, it is my great pleasure to invite you to participate, not only in the technical program of SEKE 2016 and its rich assortment of activities, but also in enjoying the beautiful San Francisco Bay Area.

This year, we received 225 submissions from 29 countries. Through a rigorous review process where a majority (90 percent) of the submitted papers received three reviews, and the rest with two reviews, we were able to select 68 full papers for the general conference (30 percent), and 46 short papers (20 percent). Out of that, 7 papers have been accepted for the DBKE workshop, and 114 papers are scheduled for presentation in forty one sessions during the conference. In addition, the technical program includes two excellent keynote speeches, 4 posters and 9 demo presentations, as well as the workshop on Big Data Research and Development in Knowledge Engineering (BDKE).

The high quality of the SEKE 2016 technical program would not have been possible without the tireless effort and hard work of many individuals. First of all, we would like to express my sincere appreciation to all the authors whose technical contributions have made the final technical program possible. We are very grateful to all the Program Committee members whose expertise and dedication made my responsibility that much easier. Our gratitude also goes to the keynote speakers who graciously agreed to share their insight on important research issues, to the conference organizing committee members for their superb work, and to the external reviewers for their contribution.

Personally, we owe a debt of gratitude to a number of people whose help and support with the technical program and the conference organization are unflinching and indispensable. We are deeply indebted to Dr. S. K. Chang, Chair of the Steering Committee, for his constant guidance and support that are essential to pull off SEKE 2016. Our heartfelt appreciation goes to Dr. Haiping Xu, University of Massachusetts Dartmouth, USA, the Conference Chair, for his help and experience, and to the Program Committee Co-Chairs, Dr. Shihong Huang, Florida Atlantic University, USA, Dr. Oscar Pereira, University of Aveiro, Portugal, Angelo Perkusich, Federal University of Campina Grande, Brazil, for their outstanding team work. In addition, we are truly grateful to the workshop organizer, Zheng Xu, Tsinghua University, China, and his team for their great job in organizing the Workshop on Big Data Research and Development in Knowledge Engineering. We also like to express our appreciation to Prof. Yong Qin, Beijing JiaoTung University, for his excellent job in organizing the special session on Transportation System Research and Development in Knowledge Engineering.

Moreover, we would like to express my great appreciation to all of the conference organization committee members, including the Publicity Chair, Dr. Robert Heinrich, Karlsruhe Institute of Technology, Germany, Demo & Poster Session Co-Chairs, Dr. Guowei Yang, Texas State University, USA and Dr. Mukul Prasad, Fujitsu Laboratories of America, USA. Moreover, we would like to appreciate and recognize our Conference Liaisons in different regions for their important contributions. They are: America Liaison - Hironori Washizaki, Waseda University, Japan, USA; Europe Liaison - Raul Garcia Castro, Universidad Politecnica de Madrid, Spain; India Liaison - Swapan Bhattacharya, National Institute of Technology Karnataka, Surathakl; and South America Liaison - Jose Carlos Maldonado, ICMC-USP, Brazil.

Last but certainly not the least, we must acknowledge the important contributions that the KSI staff members have made. Their timely and dependable support and assistance throughout the entire process have been truly remarkable. Finally, we wish you have productive discussion, great networking, effective presentation, and pleasant stay and travel in San Francisco to participate in SEKE 2016.

Jerry Gao, San Jose State University, USA, Program Committee Chair
Shihong Huang, Florida Atlantic University, USA, Program Committee Co-Chair
Oscar Pereira, University of Aveiro, Portugal, Program Committee Co-Chair
Angelo Perkusich, Federal University of Campina Grande, Brazil, Program Committee Co-Chair

SEKE 2016

The 28th International Conference on Software Engineering & Knowledge Engineering

July 1– 3, 2016

Hotel Sofitel, Redwood City, San Francisco Bay, USA

Conference Organization

CONFERENCE CHAIR

Haiping Xu, University of Massachusetts Dartmouth, USA

PROGRAM COMMITTEE CHAIR

Jerry Gao, San Jose State University, USA

PROGRAM COMMITTEE CO-CHAIRS

Shihong Huang, Florida Atlantic University, USA
Oscar Pereira, University of Aveiro, Portugal
Angelo Perkusich, Federal University of Campina Grande, Brazil

STEERING COMMITTEE CHAIR

Shi-Kuo Chang, University of Pittsburgh, USA

STEERING COMMITTEE

Vic Basili, University of Maryland, USA
Bruce Buchanan, University of Pittsburgh, USA
C. V. Ramamoorthy, University of California, Berkeley, USA

ADVISORY COMMITTEE

Jerry Gao, San Jose State University, USA
Swapna Gokhale, University of Connecticut, USA
Natalia Juristo, Universidad Politecnica de Madrid, Spain
Taghi Khoshgoftaar, Florida Atlantic University, USA
Guenther Ruhe, University of Calgary, Canada
Masoud Sadjadi, Florida International University, USA
Du Zhang, California State University, USA

PROGRAM COMMITTEE

Silvia Teresita Acuna, Universidad Autonoma de Madrid, Spain
Shadi Alawneh, C-CORE, Canada
Izzat Alsmadi, Boise State University, USA
Taisira Al-Belushi, Sultan Qaboos University, Oman
Mark Allison, University of Michigan - Flint, USA
John Anvik, Univ. of Lethbridge, Canada
Omar El Ariss, Penn State Univ at Harrisburg, USA
Doo-Hwan Bae, Korea Advanced Institute of Science and Technology, Korea
Ebrahim Bagheri, Ryerson University, Canada
Hamid Bagheri, George Mason University and Massachusetts Institute of Technology, USA
Xiaoying Bai, Tsinghua University, China
Fevzi Belli, University of Paderborn, Germany
Ateet Bhalla, Consultant, India
Swapan Bhattacharya, NITK, Surathakl, India
Alessandro Bianchi, University of Bari, Italy
Ivo Bukovsky, Czech Technical University in Prague, Czech Republic
Raul Garcia Castro, Universidad Politecnica de Madrid, Spain
Chih-Hung Chang, Hsiuping University of Science and Technology, Taiwan
Keith Chan, Hong Kong Polytechnic University, Hong Kong
Kuang-nan Chang, Eastern Kentucky University, USA
Meiru Che, University of Texas at Austin, USA
Wen-Hui Chen, National Taipei University of Technology, Taiwan
Stelvio Cimato, University of Milan, Italy
Nelly Condori-fernandez, VU University Amsterdam, The Netherlands
Fabio M. Costa, Universidade Federal de Goias, Brazil
Maria Francesca Costabile, University of Bari, Italy
Jose Luis De La Vara, Carlos III University of Madrid, Spain
Massimiliano Di Penta, University of Sannio, Italy
Scott Dick, University of Alberta, Canada
Junhua Ding, East Carolina University, USA
Fei Dong, Google Inc., USA
Derek Doran, Wright State University, USA
Weichang Du, University of New Brunswick, Canada
Christof Ebert, Vector Consulting Services, Germany
Ali Ebneenasir, Michigan Technological University, USA
Raimund Ege, NIU, USA
Magdalini Eirinaki, San Jose State University, USA
Mahmoud Elish, King Fahd University of Petroleum and Minerals, Saudi Arabia
Ruby ElKharboutly, Quinnipiac University, Canada
Davide Falessi, Cal Poly, USA
Behrouz Far, University of Calgary, Canada
Liana Fong, IBM, USA
Ellen Francine Barbosa, University of Sao Paulo, Brazil
Fulvio Frati, University of Milan, Italy
Kehan Gao, Eastern Connecticut State University, USA
Felix Garcia, University of Castilla-La Mancha, Spain
Ignacio Garcia Rodriguez De Guzman, University of Castilla-La Mancha, Spain
Swapna Gokhale, Univ. of Connecticut, USA
Wolfgang Golubski, Zwickau University of Applied Sciences, Germany
Anurag Goswami, North Dakota State Univ., USA
Desmond Greer, Queen's University Belfast, United Kingdom
Christiane Gresse Von Wangenheim, UFSC - Federal University of Santa Catarina, Brazil
Katarina Grolinger, University of Western Ontario, Canada
Hassan Haghghi, Shahid Beheshti University, Iran
Hao Han, National Institute of Informatics, Japan
Xudong He, Florida International University, USA
Robert Heinrich, Karlsruhe Institute of Technology, Germany
Miguel Herranz, University of Alcalá, Spain
Rubing Huang, Jiangsu University, China

Shihong Huang, Florida Atlantic University, USA
 Hamdy Ibrahim, University of Calgary, Canada
 Bassey Isong, North-West University, South Africa
 Clinton Jeffery, University of Idaho, USA
 Jason Jung, Chung-Ang University, South Korea
 Selim Kalayci, Florida International University, USA
 Ananya Kanjilal, B.P. Poddar Institute of Technology and Management, India
 Eric Kasten, Michigan State University, USA
 Taghi Khoshgoftaar, Florida Atlantic University, USA
 Claudiu V. Kifor, Lucian Blaga University of Sibiu, Romania
 Jun Kong, North Dakota State University, USA
 Aneesh Krishna, Curtin University of Technology, Australia
 Vinay Kulkarni, Tata Consultancy Services, India
 Jeff Lei, University of Texas at Arlington, USA
 Meira Levy, Shenkar College of Engineering and Design, Israel
 Bixin Li, Southeast University, China
 Xin Li, Google Inc., USA
 Yuan-Fang Li, Monash University, Australia
 Zhi Li, Guangxi Normal University, China
 Jianhua Lin, Eastern Connecticut State University, USA
 Lan Lin, Ball State University, USA
 Xiaoqing Frank Liu, Missouri University of Science and Technology, USA
 KaiKai Liu, San Jose State University, USA
 Shih-hsi Liu, California State University, Fresno, USA
 Ting Liu, Xian Jiaotong University, China
 Xiaodong Liu, Edinburgh Napier University, United Kingdom
 Yi Liu, Georgia College & State University, USA
 Luanna Lopes Lobato, Federal University of Goias, Brazil
 Baojun Ma, Beijing University of Posts and Telecommunications, China
 Ivan Machado, Federal University of Bahia, Brazil
 Marcelo de Almeida Maia, Federal University of Uberlândia, Brazil
 Beatriz Marin, Universidad Diego Portales, Chile
 Riccardo Martoglia, University of Modena and Reggio Emilia, Italy
 Santiago Matalonga, Universidad ORT Uruguay, Uruguay
 Hong Mei, Peking University, China
 Hsing Mei, Fu Jen Catholic University, Taiwan
 Andre Menolli, Universidade Estadual do Norte do Parana (UENP), Brazil
 Ali Mili, NJIT, USA
 Alok Mishra, Atilim University, Turkey
 Manuel Mora, Autonomous University of Aguascalientes, Mexico
 Hiroyuki Nakagawa, Osaka University, Japan
 Kia Ng, ICSRiM - University of Leeds, United Kingdom
 Allen Nikora, Jet Propulsion Laboratory, USA
 Amjad Nusayr, University of Houston-Victoria, USA
 Edson A. Oliveira Jr., State University of Maringa, Brazil
 Xin Peng, Fudan University, China
 Oscar Pereira, University of Aveiro, Portugal
 Antonio Piccinno, University of Bari, Italy
 Alfonso Pierantonio, University of L'Aquila, Italy
 Daniel Plante, Stetson University, USA
 Rick Rabiser, Johannes Kepler University, Austria
 Filip Radulovic, Universidad Politecnica de Madrid, Spain
 Damith C. Rajapakse, National University of Singapore, Singapore
 Rajeev Raje, IUPUI, USA
 Henrique Rebelo, Universidade Federal de Pernambuco, Brazil
 Marek Reformat, University of Alberta, Canada
 Robert Reynolds, Wayne State University, USA
 Elder Macedo Rodrigues, Pontifical Catholic University of Rio Grande do Sul, Brazil
 Daniel Rodriguez, Universidad de Alcala, Spain
 Ivan Rodero, The State University of New Jersey, USA

Samira Sadaoui, University of Regina, Canada
Masoud Sadjadi, Florida International University, USA
Claudio Sant'Anna, Universidade Federal da Bahia, Brazil
Abdelhak-Djamel Seriai, University of Montpellier 2 for Sciences and Technology, France
Andreas Schoenberger, Siemens AG, Germany
Michael Shin, Texas Tech University, USA
Martin Solari, Universidad ORT Uruguay, Uruguay
Qinbao Song, Xi'an Jiaotong University, China
George Spanoudakis, City University London, United Kingdom
Jing Sun, University of Auckland, New Zealand
Yanchun Sun, Peking University, China
Gerson Sunye, University of Nantes, France
Chuanqi Tao, Nanjing University of Science and Technology, China
Jeff Tian, Southern Methodist University, USA
Mark Trakhtenbrot, Holon Institute of Technology, Israel
Peter Troeger, TU Chemnitz, Germany
T. H. Tse, The University of Hong Kong, Hong Kong
Burak Turhan, Oulu University, Finland
Christelle Urtado, LGI2P Ecole des Mines d'Ales, France
Sylvain Vauttier, Ecole des mines d'Ales, France
Silvia Vergilio, Federal University of Parana (UFPR), Brazil
Gennaro Vessio, University of Bari, Italy
Sergiy Vilkomir, East Carolina University, USA
Aaron Visaggio, University of Sannio, Italy
Arndt Von Staa, Pontifical Catholic University of Rio de Janeiro, Brazil
Huanjing Wang, Western Kentucky University, USA
Jiacun Wang, Monmouth University, USA
Xiaoyin Wang, University of Texas at San Antonio, USA
Ye Wang, Zhejiang Gongshang University, China
Yong Wang, New Mexico Highlands University, USA
Ziyuan Wang, Nanjing University of Posts and Telecommunications, China
Hironori Washizaki, Waseda University, Japan
Victor Winter, University of Nebraska at Omaha, USA
Guido Wirtz, Bamberg University, Germany
Franz Wotawa, TU Graz, Austria
Dianxiang Xu, Boise State University, USA
Frank Weifeng Xu, Bowie State University, USA
Haiping Xu, University of Massachusetts Dartmouth, USA
Hongji Yang, Bath Spa University, United Kingdom
Huiqun Yu, East China University of Science and Technology, China
Jiang Yue, Fujian Normal University, China
Du Zhang, California State University, USA
Pengchen Zhang, Hohai University, China
Yong Zhang, Tsinghua University, China
Zhenyu Zhang, Institute of Software, Chinese Academy of Sciences, China
Jiang Zheng, ABB US Corporate Research Center, USA
Xingquan Zhu, Florida Atlantic University, USA
Eugenio Zimeo, University of Sannio, Italy

DEMO&POSTER SESSIONS CO-CHAIRS

Guowei Yang, Texas State University, USA
Mukul Prasad, Fujitsu Laboratories of America, USA

PUBLICITY CHAIR

Robert Heinrich, Karlsruhe Institute of Technology, Germany

ASIA LIAISON

Hironori Washizaki, Waseda University, Japan

EUROPE LIAISON

Raul Garcia Castro, Universidad Politecnica de Madrid, Spain

INDIA LIAISON

Swapan Bhattacharya, National Institute of Technology Karnataka, Surathakl, India

SOUTH AMERICA LIAISON

Jose Carlos Maldonado, ICMC-USP, Brazil

Keynote

Innovation In Every Part Of Our Business

Craig Stephens C.Eng FIET
Chief Engineer
Controls Research and Advanced Engineering
Ford Motor Company

Abstract

Competitive pressure, demands for reduced emissions, improved safety and fuel economy have long been engines of innovation within the automotive industry but today mobility and congestion have become increasingly important factors. Advances in connectivity, machine learning and analytics are providing the possibility of autonomous vehicles and personalized mobility technology as potential solutions. These revolutionary innovations promise to be as significant as the arrival of the automobile over 100 years ago. This presentation provides insight into this exciting and dynamic new space.

About the Speaker

Craig Stephens joined Ford Motor Company in 1987 working in Powertrain Calibration and Controls for Ford of Europe. In 1991 he moved to Dearborn, Michigan to lead a worldwide company effort to develop common Powertrain Control Algorithms and Software. A variety of other positions in P/T Controls and Software development followed. Following a short spell in Visteon developing Electronic Throttle Controls he returned to Ford, first in Powertrain Controls integration and then leading an advanced project for a Mild Hybrid Powertrain. From 2001 to 2009 he was Manager of Powertrain Controls in Ford's Research and Advanced Engineering activity. He held the position of Assistant Chief Engineer for Global Powertrain Control Systems until 2012 and is now Chief Engineer for Controls Engineering in Research and Advanced Engineering. In this capacity he leads a group which is involved in every aspect of automotive controls from individual features through to complex vehicle level control systems required for electrified products and autonomous driving. In addition to developing features for new products the team is also very active in developing the underlying engineering techniques in Systems, Software, Model Based Design, Functional Safety, etc to deliver these features successfully.

Keynote

Linear Software Models: An Algebraic Theory of Software Composition

Iaakov Exman
The Jerusalem College of Engineering – JCE – Azrieli
Jerusalem, Israel

Abstract

The central goal of our software theory is to solve the composition problem of a software system of any size from sub-systems down to indivisible components either produced in house, or purchased from other manufacturers. The theory guides the gradual or agile design of the system by means of quantitative criteria for the current design quality, and by highlighting problematic coupling spots to be resolved in order to improve the software design. The theory is based upon solid results of linear algebra, and the broadly accepted wisdom that modularity is essential for understandable and maintainable software design.

The main algebraic structure of the theory is the Modularity Matrix. It links structors – a generalization of classes – in the matrix columns, to their provided functionals – a generalization of methods – in the matrix rows. The theory predicts that neat modules for any given system are the blocks of a block-diagonal standard matrix. The matrix itself is the source of quantitative criteria for design quality. Module sizes are determined by the matrix eigenvectors. The theory formalizes intuitive concepts widely used in software engineering, providing precise definitions for coupling, cohesion, modules, single responsibility and canonical systems such as design patterns. This talk also touches recent research results showing that apparently different alternative theories, such as the Modularity Lattice from FCA (Formal Concept Analysis) and approaches using the Laplacian Matrix are equivalent to the Modularity Matrix. Therefore, Linear Software Models are proposed as a coherent unified algebraic theory of software composition.

About the Speaker

Iaakov Exman is a faculty member of The Jerusalem College of Engineering (JCE – Azrieli), Department of Software Engineering. He received his Ph.D. from The Hebrew University of Jerusalem, Israel (HUJI) and performed post-doctoral research at Stanford University, California, USA. He had extensive industrial experience both with very large R&D teams within IAI (Israel Aerospace Industries) and small agile groups in start-ups, and has been a research associate at HUJI, before joining JCE in 2001. His research interests are in the area of software theory, including algebraic software theory, automated software generation from ontological conceptual structures and software aspects of quantum computation. He has authored and coauthored about 100 papers in these fields. Dr. Exman has organized together with Spanish colleagues from UC3M (University Carlos III of Madrid) the SKY International Workshop on Software Knowledge, since 2010, which has been held annually in various European locations. He is actively associated with the GTSE Workshop on General Theory of Software Engineering from its first 2012 edition at the KTH in Stockholm, and since then annually co-located with the ICSE International Conference on Software Engineering.

Table of Contents

Foreword	iii
Conference Organization	iv
KeyNote 1: Innovation In Every Part Of Our Business	
Mr Craig Stephens	ix
KeyNote 2: Linear Software Models: An Algebraic Theory of Software Composition	
Prof. Iakov Exman	x
<hr/>	
Component-based Software Production	
<hr/>	
Toward Recovering Component-based Software Product Line Architecture from Object-Oriented Product Variants	1
<i>Hamzeh Eyal-Salman and Abdelhak Seriai</i>	
Improving the Applicability of Bayesian Networks through Production Rules	8
<i>Raissa Da Silva, Mirko Perkusich, Renata Saraiva, Arthur Freire, Hyggo Almeida and Angelo Perkusich</i>	
Component-based process for modeling language evaluation (S)	14
<i>Khaoula Sayeb, Oualid Khayati and Naoufel Kraeim</i>	
<hr/>	
Big Data Analysis	
<hr/>	
Keyword Search over Graph-structured Data for Finding Effective and Non-redundant Answers	18
<i>Chang-Sup Park</i>	
Informed and Timely Business Decisions – A Data-driven Approach	24
<i>Veera Tadikonda and Daniela Rosca</i>	
TACE: A Toolkit for Analyzing Concept Evolution in Computing Curricula	31
<i>Luo Tiejian, Zhang Libo, Yang Lin and Chen Xin</i>	
<hr/>	
DBRDKE-I	
<hr/>	
Sentiment Analysis of Name Entity for Text	37
<i>Xinzhi Wang, Hui Zhang, Jiayue Wang and Yang Zhou</i>	
Event Multiple Influence Calculation and Relationship of Multiple Influence Discovery (S)	42
<i>Yunlan Xue, Lingyu Xu and Jie Yu</i>	
Secure Outsourcing Algorithm of Polynomials in Cloud Computing	46
<i>Xianlin Zhou, Yong Ding, Zhao Wang, Xiumin Li, Jun Ye and Zheng Xu</i>	
Deep Data Anaylizing Application Based on Scale Space Theory in Big Data Environment	52
<i>Ye Hu, Kun Gao and Zheng Xu</i>	
<hr/>	
Open Source Software Production	
<hr/>	

Distribution and Continuity of Developers' Contributions in OSS Projects: A Case Study. 58 <i>Zhongjie Wang, Dewayne Perry and Xiaofei Xu</i>	
A Two Phase Case Study on Implementation of Open Source Development Practices within a Company Setting 64 <i>Alma Orucevic-Alagic and Martin Höst</i>	
A Novel Open Source Software Ecosystem: From a Graphic Point of View and Its Application (S) 71 <i>Chenxi Song, Tao Wang, Gang Yin, Xunhui Zhang and Cheng Yang</i>	

Agile Software Development

An FCM-based Personalized Affective Model for Agile Software Development..... 75 <i>Wenjing He, Jun Lin, Xinjia Yu, Zhiqi Shen and Chunyan Miao</i>	
A Method to Build Bayesian Networks based on Artifacts and Metrics to Assess Agile Projects..... 81 <i>Renan Willamy, João Nunes, Mirko Perkusich, Arthur Freire, Renata Saraiva, Hyggo Almeida and Angelo Perkusich</i>	
CPDScorer: Modeling and Evaluating Developer Programming Ability across Software Communities..... 87 <i>Weizhi Huang, Wenkai Mo, Beijun Shen, Yu Yang and Ning Li</i>	
Context and Trust Aware Workflow Oriented Access Framework..... 93 <i>Tapalina Bhattasali, Nabendu Chaki, Rituparna Chaki and Khalid Saeed</i>	

Big Data, Query and Management

Integrated Data Management System for Data Center..... 99 <i>Debasis Dash, Juthika Dash, Younghee Park and Jerry Gao</i>	
A Model-Driven Approach to Generate Relevant and Realistic Datasets (S)..... 105 <i>Adel Ferdjoukh, Eric Bourreau, Annie Chateau and Clémentine Nebut</i>	
A Query Language of Data Provenance Based on Dependency View for Process Analysis (S)..... 110 <i>Xuan Sun, Xin Gao, Huiying Du and Wei Ye</i>	
Collecting Usage Data for Software Development: Selection Framework for Technological Approaches 114 <i>Sampo Suonsyrjä, Kari Systä, Tommi Mikkonen and Henri Terho</i>	

DBRDKE-II

Deep Data Analyzing Method Based on Scale Space Theory (S)..... 120 <i>Ye Hu, Guangqun Chen and Kun Gao</i>	
DFIPS: Toward Distributed Flexible Intrusion Prevention System in Software Defined Network 124 <i>Xuesong Jia, Danni Ren, Yitao Yang, Huakang Li and Guozi Sun</i>	

Knowledge Flow-based Semantic Organization of Resources about Foreign Language Learning	128
<i>Li Li</i>	

Learning and Education

Self-learning Change-prone Class Prediction	134
<i>Meng Yan, Mengning Yang, Chao Liu and Xiaohong Zhang</i>	
Generating Summarized Preview for Education Resource based on Exploring and Comparing GUIs	141
<i>Chao Ma, Xiangping Chen, Yongsheng Rao and Mouguang Lin</i>	
Learning to Discover Subsumptions between Software Engineering Concepts in Wikipedia	147
<i>Xiang Dong, Kai Chen, Jiangang Zhu and Beijun Shen</i>	
How Instructional Feedback Has Been Employed in Instructional Units for Teaching Software Project Management Tools: A Systematic Literature Review	153
<i>Rafael Gonçalves and Christiane Gresse von Wangenheim</i>	

Defect Prediction and Analysis I

A Ranking-Oriented Approach to Cross-Project Software Defect Prediction: An Empirical Study	159
<i>Guoan You and Yutao Ma</i>	
Stage-oriented Analysis on Factors Impacting Bug Fixing Time	165
<i>Hong Wu, Junjie Wang, Qing Wang, Lin Shi and Feng Yuan</i>	
Heterogeneous Defect Prediction via Exploiting Correlation Subspace	171
<i>Ming Cheng, Guoqing Wu, Min Jiang, Hongyan Wan, Guoan You and Mengting Yuan</i>	
Long-Term Active Integrator Prediction in the Evaluation of Code Contributions	177
<i>Jing Jiang, Fuli Feng, Xiaoli Lian and Li Zhang</i>	

AI and Knowledge Engineering

TDR System - A Multi-Level Slow Intelligence System for Personal Health Care	183
<i>Shikuo Chang</i>	
Building a Domain Knowledge Base from Wikipedia: a Semi-supervised Approach	191
<i>Kai Chen, Xiang Dong, Jiangang Zhu and Beijun Shen</i>	
A Knowledge Modeling Framework for Computational Materials Engineering (S)	197
<i>Raghavendra Reddy Yeddula, Sushant Vale, Sreedhar Reddy, Chetan P Malhotra, Gautham B. P. and Pramod Zagade</i>	

DBRDKE-III

Building Ontology for Different Emotional Contexts and Multilingual Environment in Opinion Mining (S)	203
<i>Wan Tao, Tao Liu and Wei Yu</i>	

Building the Multi-layer Theory of Association Semantic based on the Power-law Distribution of Linking Keywords	207
<i>Guangli Zhu, Xiaojun Tang, Shunxiang Zhang and Zheng Xu</i>	
Developer Recommendation with Awareness of Accuracy and Cost	213
<i>Jin Liu, Yiqiuzi Tian, Liang Hong, Xu Chen and Zhou Xu</i>	

Optimization

Managing forest transportation planning using ant colony optimization method	219
<i>Pengpeng Lin, Ruxin Dai and Ran An</i>	
Multi-Objective Biogeography-Based Method to Optimize Virtual Machine Consolidation	225
<i>Kai Shi, Huiqun Yu, Fei Luo and Guisheng Fan</i>	
MyBatRecommender: Automated optimization of energy consumption for Android smartphones in software layer	231
<i>Marcel Popolin de Araújo Cunha and Luciana Aparecida Martinez Zaina</i>	

Defect Prediction and Analysis II

A Multi-Source TrAdaBoost Approach for Cross-Company Defect Prediction	237
<i>Xiao Yu, Jin Liu, Mandi Fu, Chuanxiang Ma and Guoping Nie</i>	
Exploring the Influence of Time Factor in Bug Report Prioritization	243
<i>Zhengjie Xu, Tieke He, Weiqiang Zhang, Yabin Wang, Jia Liu and Zhenyu Chen</i>	
A Model for Predicting Bug Fixes in Open Source Operating Systems: an Empirical Study (S)	249
<i>Alberto Sillitti and Paolo Ciancarini</i>	

Data Analysis and Systems

Exploring DevOps for Data Analytical System with Essential Demands Elicitation	255
<i>Jiabin Zheng, Yan Liu and Jin Lin</i>	
Criminal Network Analysis with Interactive Strategies: A Proof of Concept Study using Mobile Call Logs	261
<i>Peng Zhou, Yan Liu, Mengjia Zhao and Xin Lou</i>	
Combining Smartphone and Smartwatch Sensor Data in Activity Recognition Approaches: an Experimental Evaluation	267
<i>Felipe Barbosa Araújo Ramos, Anne Lorayne, Antonio Alexandre Moura Costa, Reudismam Rolim de Sousa, Hyggo Almeida and Angelo Perkusich</i>	
Time Series Classification with Discrete Wavelet Transformed Data: Insights from an Empirical Study	273
<i>Daoyuan Li, Tegawendé F. Bissyandé, Jacques Klein and Yves Le Traon</i>	

Transportation System Research & Development

An Improved MFD based Regional Traffic Volume Dynamic Control	279
<i>Yiman Du, Jianping Wu, Yuhan Jia and Ming Xu</i>	

Traffic Safety Region Estimation Based on SFS-PCA-LSSVM: An Application to Highway Crash Risk Evaluation	285
<i>Yanfang Yang and Yong Qin</i>	
Sensor-Based Detection Approach for Passenger Flow Safety in Chinese High-speed Railway Transport Hub	291
<i>Zhengyu Xie and Yong Qin</i>	
City Freight Intelligent Scheduling Model Based on Genetic Algorithm (S)	297
<i>Sai Shao, Ailing Huang and Haijun Liu</i>	
Clustering and Artificial Neural Network Ensembles Based Effort Estimation	301
<i>Hamdy Ibrahim and Behrouz Far</i>	
<hr/> Software System Development & Specification <hr/>	
An Algorithm for Forward Reduction in Sequence-Based Software Specification	309
<i>Lan Lin and Yufeng Xue</i>	
MSECO-DEV: Application Development Process in Mobile Software Ecosystems	317
<i>Awdren Fontão, Rodrigo Santos, Jackson Feijó Filho and Arilo Claudio Dias-Neto</i>	
<hr/> Semantic Web and SOA <hr/>	
Layered Implementation View of a SOA Based Electronic Health Record	323
<i>Josimar Lima, Joyce França, Jislane Menezes, Adicinéia Oliveira and Michel Soares</i>	
An Ontology-driven Adaptive System for the Patient Treatment Management (S)	329
<i>Emna Mezghani, Marcos Da Silveira, Cédric Pruski, Ernesto Exposito and Khalil Drira</i>	
<hr/> Quality Assurance I <hr/>	
Cross-Model Traceability for Coupled Transformation of Software and Performance Models	333
<i>Nariman Mani, Dorina Petriu and Murray Woodside</i>	
Exploring the Dimensions of Electronic Government Service Quality (S)	341
<i>Tasira Al Balushi and Saqib Ali</i>	
A Systematic Mapping Study on Legacy System Modernization (S)	345
<i>Everton Agilar, Rodrigo Almeida and Edna Canedo</i>	
<hr/> System Architecture <hr/>	
Extending Architecture Description Languages With Exchangeable Component Behavior Languages	351
<i>Robert Heim, Bernhard Rumpe and Andreas Wortmann</i>	
The Software Architecture Mapping Framework for Managing Architectural Knowledge ...	357
<i>Sebastien Adam and Alain Abran</i>	
RARep: a Reference Architecture Repository	363
<i>Tales Prates Correia, Milena Guessi, Lucas Bueno Ruas Oliveira and Elisa Yumi Nakagawa</i>	

A Multi-Agent Architecture for Quantified Fruits: Design and Experience (S).....	369
<i>Jean-Pierre Briot, Nathalia Moraes Do Nascimento and Carlos José Pereira de Lucena</i>	

Software Testing I

Quality Assurance for Big Data Application – Issues, Challenges, and Needs.....	375
<i>Chuanqi Tao and Jerry Gao</i>	
Interactive Tool for Iterative Test Suite Construction (S).....	382
<i>Matthew Patrick</i>	

Machine Learning I

PRO-Fit: A personalized fitness assistant framework (S).....	386
<i>Saumil Dharia, Vijesh Jain, Jvalant Patel, Jainikkumar Vora, Shaurya Chawla and Magdalini Eirinaki</i>	
A Machine Learning Approach for Developing Test Oracles for Testing Scientific Software (S).....	390
<i>Junhua Ding and Dongmei Zhang</i>	

Cloud Computing & Services

A Systematic Mapping Study on the Multi-tenant Architecture of SaaS Systems.....	396
<i>Victor H. S. C. Pinto, Helder J. F. Luz, Ricardo R. Oliveira, Paulo S. L. Souza and Simone R. S. Souza</i>	
Applying Verbal Decision Analysis to Task Allocation in Distributed Development of Software (S).....	402
<i>Marum Simão Filho, Placido Rogério Pinheiro and Adriano Albuquerque</i>	

Software Testing II

Effectively Testing of Timed Composite Systems using Test Case Prioritization (S).....	408
<i>Huu Nghia Nguyen, Fatiha Zaidi and Ana Cavalli</i>	
Improving Accuracy of Patient Synthetic Data for Testing Medical Cyber-Physical Systems (S).....	414
<i>Leonardo Da Costa Santos, Lenardo Chaves E Silva, Ana Luisa Medeiros, Hyggo Oliveira de Almeida and Angelo Perkusich</i>	
Capture & Replay with Text-Based Reuse and Framework Agnosticism.....	420
<i>Filipe Arruda, Augusto Sampaio and Flavia Barros</i>	

Machine Learning II

Efficiently Measuring an Accurate and Generalized Clone Detection Precision using Clone Clustering.....	426
<i>Jeffrey Svajlenko and Chanchal K. Roy</i>	

NuSense: A Sensor-Based Framework for Ambient Awareness applied in Game Therapy Monitoring (S)	434
<i>Túlio Costa, Breno Lacerda A. Paiva, Rinaldo V. Menezes, Lukas T. Ramos, Andrei G. Lopes and Frederico Moreira Bublitz</i>	
<hr/> Cloud Computing & Services <hr/>	
Modeling and analyzing cost-aware fault tolerant strategy for cloud application (S)	439
<i>Liqiong Chen, Guisheng Fan and Yunxiang Liu</i>	
From Design to Code: An Educational Approach	443
<i>Candice Eckert, Brian Cham, Jing Sun and Gillian Dobbie</i>	
<hr/> Data Mining I <hr/>	
SLTM: A Sentence Level Topic Model for Analysis of Online Reviews (S)	449
<i>Yuhan Zhang and Haiping Xu</i>	
Using Frequent Closed Pattern Mining to Solve a Consensus Clustering Problem	454
<i>Atheer Al-Najdi, Nicolas Pasquier and Frederic Precioso</i>	
Mining Feature-Opinion from Reviews Based on Dependency Parsing (S)	462
<i>Rui Hao, Tieke He, Hang Qi and Jia Liu</i>	
<hr/> Software Testing III <hr/>	
Experimentation in the Industry for Automation of Unit Testing in a Business Intelligence Environment (S)	466
<i>Igor Peterson Oliveira Santos, Andre Nascimento, Juli Kelle Gois Costa, Methanias Colaço Júnior and Wenderson Campos Pereira</i>	
Automatically Finding Hidden Industrial Criteria used in Test Selection (S)	470
<i>Cláudio Magalhães, Alexandre Mota and Eliot Maia</i>	
Practical Combinatorial Testing Approaches: A Case Study of a University Portal Application (S)	474
<i>Mary Frances Moore and Sergiy Vilkomir</i>	
<hr/> Software Testing IV <hr/>	
On Building Test Automation System for Mobile Applications Using GUI Ripping	480
<i>Chuanqi Tao and Jerry Gao</i>	
Redroid: A Regression Test Selection Approach for Android Applications	486
<i>Quan Do, Guowei Yang, Meiru Che, Darren Hui and Jefferson Ridgeway</i>	
Conformance Testing of Balana: An Open Source Implementation of the XACML3.0 Standard	492
<i>Sung-Ju Fan Chiang, Daniel Chen and Dianxiang Xu</i>	
<hr/> Data Mining II <hr/>	
QCC:A novel cluster algorithm based on Quasi-Cluster Centers	498
<i>Huang Jinlong, Zhu Qingsheng, Yang Lijun and Cheng Dongdong</i>	

Information Mining Projects Management Process (S).....	504
<i>Sebastian Martins, Patricia Pesado and Ramon Garcia-Martinez</i>	
An investigation of students behavior in discussion forums using Educational Data Mining (S).....	510
<i>Crystiano José Richard Machado, Bruno Rafael B. Lima, Alexandre Maciel and Rodrigo Lins Rodrigues</i>	
<hr/> Quality Assurance II <hr/>	
Empirical Evaluation of the ProcessPAIR Tool for Automated Performance Analysis.....	515
<i>Mushtaq Raza, João Faria and Rafael Salazar</i>	
Embedded Emotion-based Classification of Stack Overflow Questions Towards the Question Quality Prediction (S).....	521
<i>Amit Kumar Mondal, Mohammad Masudur Rahman and Chanchal K. Roy</i>	
Multi-Objective Optimization for Software Testing Effort Estimation (S).....	527
<i>Solomon Mensah, Jacky Keung, Kwabena Ebo Bennin and Michael Franklin Bosu</i>	
<hr/> Requirement Engineering I <hr/>	
Effectiveness of Human Error Taxonomy during Requirements Inspection: An Empirical Investigation.....	531
<i>Vaibhav Anu, Gursimran Walia, Wenhua Hu, Jeffrey Carver and Gary Bradshaw</i>	
Requirements Engineering Related Usability Techniques Adopted in Agile Development Processes.....	537
<i>Daniel A. Magües, John W. Castro and Silvia T. Acuña</i>	
<hr/> Green Computing & Analysis <hr/>	
Towards Optimization of Energy Consumption of Drones with Software-Based Flight Analysis (S).....	543
<i>Ilenia Fronza, Luis Corral, Nabil El Ioini and Aristeia Ibershimi</i>	
Choosing the Best Strategy for Energy Aware Building System: an SVM-based Approach (S).....	547
<i>Yuanyang Wang, Xiaohong Chen, Haiying Sun and Mingsong Chen</i>	
An Adaptable Approach for Indoor Location (S).....	551
<i>Mário Melo and Gibeon Aquino</i>	
<hr/> Project Management & Content Management <hr/>	
Improving risk identification process in project management (S).....	555
<i>Allan G. Contessoto, Laís A. Sant' Ana, Rogéria C. G. Souza, Carlos R. Valêncio, Geraldo F. D. Zafalon, Anderson R. Amorim and Antonio M. N. Esteca</i>	
On Criteria to Choose a Content Management System: A Technology Acceptance Model Approach.....	559
<i>Jislane Silva, Danilo Ramos and Michel Soares</i>	

Global Software development: An approach to design and evaluate the risk factors for global practitioners (S).....	565
<i>Chamundeswari Arumugam and Baskaran Kaliamourthy</i>	

Requirement Engineering II

Problem-Aware Traceability in Goal-Oriented Requirements Engineering (S).....	569
<i>Grace Park, Lawrence Chung, Jang-Eui Hong, José Luis Garrido and Manuel Noguera</i>	
Approach to Define a Non-Functional Requirements Elicitation Guide Using a Customer Language (S)	575
<i>Andreia Silva, Placido Pinheiro, Adriano Albuquerque and Jonatas Barroso</i>	
A Characterization of Negative User Stories (S)	579
<i>Pankaj Kamthan and Nazlie Shahmir</i>	

Web System Study & Analysis

Quantifying and Assessing the Merge of Cloned Web-Based System: An Exploratory Study.....	583
<i>Jadson Santos and Uirá Kulesza</i>	
Automaticly Generating Web Page From A Mockup (S)	589
<i>Ruozi Huang, Yonghao Long and Xiangping Chen</i>	

Empirical Study & Social Network Analysis

An Empirical Study to Evaluate the Feasibility of a UX and Usability Inspection Technique for Mobile Applications (S).....	595
<i>Ingrid Costa, Williamson Silva, Adriana Lopes, Luis Rivero, Bruno Gadelha, Elaine Oliveira and Tayana Conte</i>	
How Novice Software Engineers Apply User Interface Design Patterns: An Empirical Study (S).....	600
<i>Luis Jorge Enrique Rivero Cabrejos and Tayana Conte</i>	

Software Evolution

Software Evolution by Correctness Enhancement	605
<i>Wided Ghardallou, Nafi Diallo and Ali Mili</i>	
Risk Management Analysis in Software Projects which Use the Scrum Framework (S)	611
<i>Breno Gontijo Tavares, Carlos Eduardo Sanches Da Silva and Adler Diniz de Souza</i>	
DmS-Modeler: A Tool for Modeling Decision-making Systems for Self-adaptive Software Domain	617
<i>Frank Affonso, Gustavo Leite and Elisa Nakagawa</i>	

Modeling & Analysis

Modeling and Analyzing Security Patterns Using High Level Petri Nets (S).....	623
<i>Xudong He and Yujian Fu</i>	

Evaluating the Representation of User Interface Elements in Feature Models: an Empirical Study (S).....	628
<i>Ildevana Poltronieri Rodrigues, Ana Paula Bacelo, Milene Selbach Silveira, Marcia de Borba Campos and Elder de Macedo Rodrigues</i>	
Towards a Systematic Approach to Graph Data Modeling: Scenario-based Design and Experiences.....	634
<i>Mengjia Zhao, Yan Liu and Peng Zhou</i>	

Software Engineering Methodology

An Agile Methodology for Reengineering Object-Oriented Software.....	638
<i>Anam Sahoo, David Kung and Sanika Gupta</i>	
A Content-Based Approach for Recommending UML Sequence Diagrams (S).....	644
<i>Thaciana G. O. Cerqueira, Franklin Ramalho and Leandro Balby Marinho</i>	
Software Clustering using Hybrid Multi-Objective Black Hole Algorithm (S).....	650
<i>Kawal Jeet and Renu Dhir</i>	

Poster

MABT - a multiagent-based toolkit for transforming existing systems into self-adaptive systems (P).....	654
<i>Lu Wang, Qingshan Li, Yishuai Lin and Hua Chu</i>	
An Efficient Algorithm to Identify Minimal Failure-Causing Schemas from Exhaustive Test Suite (P).....	655
<i>Yuanchao Qi, Qi Wang, Chiya Xu, Tieke He and Ziyuan Wang</i>	
A qualitative analysis of the adherence between the Information Technology Solution Acquisition Guide, for Brazilian Federal Public Administration and, the CMMI models (P)	657
<i>Luiz Sergio Placido Da Silva, Renata Moreira, Alexandre Marcos Lins de Vasconcelos, Mauricio Ronny Souza and Suzana Cândido de Barros Sampaio</i>	
Early Detection of Suicide Using Big-Data Analytics in Real Time (P).....	659
<i>Hardik Patel and Cheng-Yuan Hsieh</i>	

Authors' Index	A-1
Program Committee Reviewers' Index	A-10
External Reviewers' Index	A-14

Note:

(S) indicates a short paper.

(P) indicates a poster.

Toward Recovering Component-based Software Product Line Architecture from Object-Oriented Product Variants

Hamzeh Eyal-Salman, Abdelhak-Djamel Seriai

UMR CNRS 5506, LIRMM, University of Montpellier 2 for Sciences and Technology, France

E-mail: {Eyalsalman, Seriai}@lirmm.fr

Abstract

Usually, companies meet different customer needs in a particular domain by developing variants of a software product. This is often performed by ad-hoc copying and modifying of various existing variants to fit purposes of new one. As the number of product variants grows, such an ad-hoc development causes severe problems to maintain these variants. Software Product Line Engineering (SPLE) can be helpful here by supporting a large-scale reuse systematically. SPL architecture (SPLA) is a key asset as it is used to derive architecture for each product in SPL. Unfortunately, developing SPLA from scratch is a costly task. In this paper, we propose an approach to contribute for recovering SPLA from existing product variants. This contribution is two-fold. Firstly, identifying common features and variation points of features of a given collection of product variants. Secondly, exploiting commonality and variability in terms of features to identify mandatory components and variation points of components as an important step in this recovering process. To evaluate the proposed approach, we applied it to two case studies. The experimental results bring evidence the effectiveness of our approach.

Keywords: product variants, software product line architecture, component, variability, feature, recovering.

1 Introduction

It is common for companies developing variants of a software product to accommodate different customer needs. These variants provide common features and differ from one another by providing unique feature combinations. A feature is “a prominent or distinctive user-visible aspect, quality or characteristic of a software system or systems” [1]. Companies develop separate product variants where a new product is built by ad-hoc copying and modifying of various existing variants to fit purposes of new one. Separately maintaining these variants causes challenges. Changes in the source code corresponding to common features (e.g., for bug fixing) must be repeated across

product variants. Therefore, companies need to change their software development strategy by a transition to a systemic reuse approach, such as Software Product Line Engineering (SPLE) to avoid such maintenance problems.

Reuse is a main characteristic of SPLE. It builds core assets consisting of all reusable software artifacts. SPL architecture (SPLA) is a key asset [2]. It is a core architecture that captures high level design decisions for SPL products, including the variation points and variants. These decisions concern in the organization of components and general rules that these components have to obey. Commonality and variability in SPLA (i.e., mandatory components and variation points (VPs) of components) globally originate from commonality and variability of customers’ requirements/features which are represented by feature model [3].

Developing SPLA from scratch is a costly task because it should encompass the components realizing all the mandatory and varying features in a particular domain. Therefore, it is useful to exploit product variants for reducing the development cost. Recovering SPLA from software product variants is not considered in the literature. Existing works support recovering software architecture from single existing software system [4]. In this paper, we present an approach to contribute for recovering SPLA from existing product variants. Our contribution is two-fold. Firstly, identifying common features and VPs of features as this organization of features represents the main source of commonality and variability in SPLA. Secondly, exploiting such identification to identify mandatory components and VPs components for SPLA. We adapt our previous component extraction approach (*ROMANTIC*) approach to extract components [5].

The remainder of the paper is organized as follows: we give a necessary background in section 2. Section 3 detail the proposed approach steps. Next, section 4 shows experimental results and analysis. Sections 5 and 6 discuss the related work and conclude the paper, respectively.

2 Variation Points in Feature Model and Software Product Line Architecture

Development of SPLs mainly relies on exploiting commonality and managing the variability between SPL's products to meet all customer requirements [3]. Feature Model (FM) is a well-known artifact to represent this commonality and variability in terms of features. Commonality refers to feature(s) that are a mandatory part of each product. FM offers three types of feature groups: *XOR-Group* (alternative), *OR-Group* and *AND-Group* (see Figure 1). Also, we consider all optional features fallen down from FM root as a single feature group called *OP-Group*. Each feature group represent a VP at the feature level. Such a VP is reflected in SPLA as a VP of components. An example to clarify the concept of VP in both SPLA and FM is shown in Figure 2. This figure shows all available alternatives for a customer to choose a phone that only supports *color*, *high resolution* or *basic* screen. It shows that each screen option is realized by a combination of two components and selection the appropriate combination is based on customer choose. Therefore, an explicit link between features and components is needed to bind variability in SPLA (VPs) according to customers' needs.

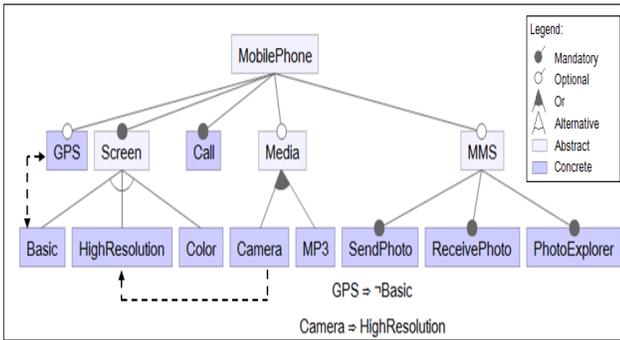


Figure 1: FM of MobilePhone-SPL [6].

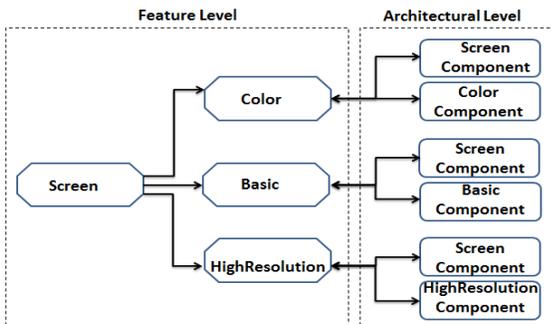


Figure 2: An Example of a Variation Point at Feature and Architecture Levels.

In our previous work [5], we proposed ROMANTIC approach to automatically recover a component-based architecture from the source code of a single existing object-oriented software. Component is a cluster of classes collaborating to provide a function of a software system. In this paper, we reuse this approach to extract components from the implementation of each feature group (VP) and common features.

3 The Proposed Approach

As SPLA encompasses commonality and variability of customers' requirements/features, we should first identify mandatory features (commonality) and VPs of features (variability) across product variants. Then, we need to exploit this commonality and variability in SPLA as mandatory components and VPs of components respectively. In our approach, components are extracted from the implementation of each group of features (i.e., mandatory features group and VPs).

Our proposed approach is organized into two phases. In the first phase, we identify mandatory features and VPs of features. This phase takes as input product configurations and feature descriptions. In the second phase, we identify mandatory components and VPs of components. This phase takes as input mandatory features, VPs of features and all feature implementations (as source code classes). This phase starts by extracting components from classes of each VP of features and from mandatory features group. Next, each feature is linked to its corresponding component(s) for binding commonality and variability at the architectural level in order to identify mandatory components and VP of components.

3.1 Identifying Mandatory Features and Variation Points of Features

We analyze the behavior of members of each VP across product configurations of a given collection of product variants. A product configuration is a combination of features supported by a product. This behavior represents a pattern of VPs. We propose algorithms to detect VP pattern by considering the definition of each type of VPs.

3.1.1 Basic Definitions

To explain our proposed algorithms, we use a FM inspired from the mobile phone industry (see Figure 1) to generate valid configurations [6]. We treat the generated configurations as product variants configurations. Table 1 shows all possible valid configurations that can be generated from FM in Figure 1. The check symbol (✓) refers to the features

provided by each configuration. Before presenting our algorithms, we start by defining the key concepts which are shared among the proposed algorithms.

Definition 1 (Feature List). *Feature list (FL) is a list of all unique features in all product configurations. All features provided by FM shown in Figure 1 represent an example of FL.*

Definition 2 (Feature Set). *Feature set is a tuple $FS = [sef, \overline{sef}]$ where sef and \overline{sef} are respectively the set of selected and not-selected features of a product [7]. Thus $sef \cap \overline{sef} = \Phi$ and $sef \cup \overline{sef} = FL$. The $P.sef$ and $P.\overline{sef}$ respectively refer to the set of selected and not-selected features of a product P .*

An example of a FS is product $P_{18} = [\{Call, SendPhoto, ReceivePhoto, PhotoExplorer, HighResolution, GPS\}, \{Basic, Color, Camera, MP3\}]$ in Table 1.

Definition 3 (Feature Set Table). *Feature set table (FST) is a collection of FS s. Each row in this table represents a product so that for every product P_i we have $P_i.sef \cup P_i.\overline{sef} = FL$. Table 1 is an example of FST.*

3.1.2 Identifying Mandatory Features

Mandatory features can be identified simply by comparing feature names of given product configurations (FST) to find features that are part of each configurations. Then, we prune FST and FL by excluding mandatory features from their contents.

3.1.3 Identifying AND Variation Points of Features

The behavior of an AND-Group of features across product variants seems like an atomic set that its members always appear or disappear together.

Based on this regular behavior, we start by intersecting pair-wisely all selected feature sides ($P_i.sef$) for all products in FST . This intersection aims at finding candidate sets that their members (features) may appear together. Of course, not each set represents AND-Group. Therefore, we pair-wisely check the members of each set against the semantic of the AND-Group. Any pair that does not respect this semantic is rejected. Consequently, each resulted set is either an AND-Group only consisting of two members or a pair of features that comply to a *require* constraint like *HighResolution* and *Camera* features in Figure 1. Therefore, we use feature descriptions to filter out pairs that comply to a *require* constraint by conducting textual matching between terms of feature descriptions. The idea behind using feature descriptions is that features that belong to the

Table 1: Product Configurations of Mobile Phone SPL

Product	Ca	S	R	P	B	H	Co	Cam	M	G
P1	✓				✓					
P2	✓					✓				
P3	✓						✓			
P4	✓					✓				✓
P5	✓						✓			✓
P6	✓					✓		✓		
P7	✓				✓				✓	
P8	✓					✓			✓	
P9	✓						✓		✓	
P10	✓					✓		✓	✓	
P11	✓					✓		✓	✓	✓
P12	✓					✓			✓	✓
P13	✓						✓		✓	✓
P14	✓					✓		✓	✓	✓
P15	✓	✓	✓	✓	✓					
P16	✓	✓	✓	✓		✓				
P17	✓	✓	✓	✓			✓			
P18	✓	✓	✓	✓		✓				✓
P19	✓	✓	✓	✓			✓			✓
P20	✓	✓	✓	✓		✓		✓		
P21	✓	✓	✓	✓	✓				✓	
P22	✓	✓	✓	✓		✓			✓	
P23	✓	✓	✓	✓			✓		✓	
P24	✓	✓	✓	✓		✓		✓	✓	
P25	✓	✓	✓	✓				✓		✓
P26	✓	✓	✓	✓		✓			✓	✓
P27	✓	✓	✓	✓			✓		✓	✓
P28	✓	✓	✓	✓		✓		✓	✓	✓

Note: we use as column labels the shortest distinguishable prefix of the feature names (e.g. Co for Color feature shown in Figure 1)

same group should have common terms in their descriptions. As a result, the remaining pairs represent only AND-Group consisting of two features. Then, we merge together pairs that have transitive relations to form AND-Groups of three or more members. Finally, we prune FST and FL by excluding AND-Groups members.

3.1.4 Identifying XOR Variation Points of Features

The behavior a XOR-Group members across product configurations imposes that the existence only one member in $P_i.sef$ while the remaining members in $P_i.\overline{sef}$. Based on this behavior, we propose Algorithm 1 to identify XOR-Groups of features. The main data structures are used through the algorithm are *Multiset* and *HashMap* (line 1). In lines (2-6), we assume that each feature (F) in FL is a member of a XOR-Group. Therefore, if F is provided by many products in FST , we obtain many sets corresponding to F . Of course, not all elements of these sets have exclusive relations with F . Therefore, we intersect all these sets to filter out irrelevant elements (features) as much as possible. After the intersection, F corresponds to a unique set. Then, F and its corresponding set is kept as an entry in a *HashMap* called *ExRe*. Each entry represents *excluded-relation* that takes the following formate $[F \Leftrightarrow set\ of\ features]$. F represents the left-hand side (LHS) while its corresponding set represents the right-hand side (RHS). Figure 3 shows *excluded-relations* obtained from our illustrative example.

The elements of RHS of an entry in *ExRe* are a combination of features so that this combination may only consist of members of the same XOR-Group's or it may include members of other XOR-Groups. Therefore, lines (7-14) check

Algorithm 1: Identifying XOR-Group Variation Points

Input: FST, FL, Feature Descriptions
Output: XGF (XOR-Groups of Fetures)

```
1 XGF ← ∅, //multiset    ExRe ← ∅ //HashMap
2 foreach  $F \in FL$  do
3   foreach  $i$  from 1 to  $|FST|$  do
4     if  $F \in P_i.sef$  then
5        $intersect \leftarrow intersect \cap P_i.sef$ 
6      $ExRe.put(F, intersect)$  // $ExRe.put(key, value)$ 
7 foreach  $entry\ En \in ExRe$  do
8    $Set\ RHS \leftarrow En.getValue(),\ count \leftarrow 0$ 
9    $Set\ CurExRe \leftarrow RHS, add(CurExRe, En.getKey())$ 
10  if  $!(CurExRe \leftarrow XGF)$  then
11    foreach  $feature\ f \in RHS$  do
12       $Set\ temp1 \leftarrow CurExRe, remove(temp1, f)$ 
13      if  $temp1 \subseteq ExRe.getValue(f)$  then
14         $count++$ 
15    if  $count = |RHS|$  then
16       $add(XGF, CurExRe)$ 
17  $XGF \leftarrow ApplyFeaDes(XGF)$ 
18  $Purne(FST, FL, XGF)$ 
19 return XGF
```

each entry in $ExRe$ against the definition of XOR-Group. Considering that an entry (En) in $ExRe$ is a XOR-Group, this means that each feature (f) in the RHS of En must appear as a LHS of another *excluded-relation* ($Ex1$) and RHS of $Ex1$ must contain all En 's features except f . By applying these lines, entries 2, 3 and 4 in Figure 3 are identified as candidate XOR-Groups and put in XGF while others are rejected (lines 15-16). XGF may contain groups only consist of two members (e.g., entry 3 in Figure 3). Such groups are not necessary to represent XOR-Groups. They may be pairs of features that comply to an *exclude-constraint* such as *GPS* and *Basic* features in Figure 1. Therefore in line 17, we use feature descriptions ($ApplyFeaDes()$) to identify and then remove such groups from XGF. Additionally, XGF may contain groups have the semantic of a XOR-Group but in fact their members can not be aggregated as a group. Such a situation occurs due to cross tree constraints (i.e., require and exclude constraints). For example, entry 2 in Figure 3 is identified as XOR-Group in spite of *Camera* feature in this entry belongs to OR-Group. This happened due to the cross tree constraint between *Camera* and *HighResolution*, which create an alternative relation between *Camera* and both *Basic* and *Color* (see Figure 1). In this situation, we use also feature descriptions to identify such groups. In line 18, we prune FST and FL by excluding members of XOR-Groups.

1-[Basic ↔ Camera, Color, GPS, HighResolution]
2-[Camera ↔ Basic, Color]
3-[GPS ↔ Basic]
4-[HighResolution ↔ Basic, Color]
5-[Color ↔ Basic, Camera, HighResolution]

Figure 3: All Excluded-Relations of FM in Figure 1.

3.1.5 Identifying OR and OP Variation Points of Features

Although the remaining features in FST are only OR-Groups and OP-Groups, the identification of their members is a challenge because the behavior of members of these groups is arbitrarily. For the identification of OP-Groups, we completely rely on feature descriptions. We consider features that have common keywords in their descriptions belong to the same OR-Group. In this way, we can identify OR-Groups. After identifying OR-Groups, the remaining features represent the members of a single OP-Group.

3.2 Identifying Mandatory Components and Variation Points of Components

3.2.1 Component Extraction

In our approach, a component is extracted based on ROMANTIC approach as a cluster of classes. ROMANTIC is applied to source code classes implementing mandatory features, AND-Group, XOR-Group, OR-Group and OP-Group. Such an application respects the components organization in SPLA (i.e., mandatory components and VPs of components). Components that are extracted from the implementation of mandatory features constitute the mandatory components while components extracted from each feature group constitute members of a VP in SPLA.

The implementations of feature group may have shared source code classes. Such classes are not specific to a certain feature group and they implement features having cross cutting behavior across all other features. Therefore, we determine such classes and then we apply ROMANTIC to these classes as a group. The extracted components represent a group of component called *Shared-Com*. They are not specific for a certain VP in SPLA. The selection of these components for products development depends on the selection of their associated features.

3.2.2 Recovering Feature-to-Component Traceability Links

From the previous step, we notice that components of a VP are not necessary having the semantic of that VP. For example, consider that $F1$ and $F2$ are two features which belong

to XOR-Group, and components extracted from the source code implementing this group are [com1, com2, com3 and com4]. Also, assume that the first three components implement F1 while com4 implements F2. In this case the relation among the first three components is not exclusive although they belong to exclusive VP. This is because the mapping between components and features is many-to-many [3]. This means that a feature’s implementation may be scattered over more than one component inside a VP and also a component may implement more than one feature. This mapping should be considered during the creation of VPs through establishing explicit links between features and their corresponding components in SPLA. These links are useful for binding variability in SPLA. Such links determine a combination of components inside each VP so that each combination represents a variant of that VP. Determining variants of each VP in SPLA is important to specify the constraints among components.

Such traceability links between features and components can be established by exploiting the transitive relation between features and components through source code classes. A feature is implemented by classes and a component is composed of classes. Therefore, classes are a shared element between features and components, which allow linking them together. After such linking, it is normal to have shared components between all features belonging to the same VP due to the many-to-many mapping between features and components. Therefore, these components are not specific to a certain feature but they are related to the parent of those features and form a parent of VP of components in SPLA.

4 Experimental Results and Evaluation

We organize our evaluation into two parts. In the first part, we evaluate the algorithms used to identify mandatory features and VPs of features. In the second part, we evaluate the identification of mandatory components and VPs of components.

4.1 Case Studies

To evaluate our approach, we apply it to two case studies: *ArgoUML-SPL* and *MobileMedia*. *ArgoUML-SPL* is the SPL for the UML modeling tool *ArgoUML*. It is open source JAVA application and provides nine features. These features are organized as a mandatory feature, an OR-Group and an OP-Group. It supports two features (*Cognitive Support* and *Logging*) that have crosscutting behavior through all other features [8]. We obtain the description of each feature of *ArgoUML-SPL* through its official website and manual instructions¹. Due to space limitation we can not

¹<http://argouml-spl.stage.tigris.org/>

be able to present *ArgoUML-SPL*’s FM. *MobileMedia* is a JAVA open source which manipulates multimedia on mobile devices. It was implemented in 8 subsequent releases. Each release represents a variant corresponds to an evolutionary step of the system development. We only consider releases (1-3 and 5-6) due to the nature of the evolution, as features in excluded releases do not have the same implementation in these releases. The description of *MobileMedia*’s features are obtained by official website of *MobileMedia*², descriptions of its use cases and analyzing source code comments.

4.2 Validating the Identification of Mandatory Features and VPs of Features

As a base for evaluation, we match each VP identified by our approach with its corresponding VP in the focused FM. This matching is measured by using two metrics inspired from information retrieval field, namely *Precision* and *Recall*. *Precision* measures the accuracy of identifying members of a VP according to the relevant members of that VP. *Recall* measures to what degree the members of identified VP covers the relevant members of that VP. The relevant members of each VP are determined from the FM. All measures have values within [0,1]. Our proposed algorithms aims to achieve high precision and recall. We propose the equations 1 and 2 to adapt *Precision* and *Recall* in our context. *IM_VP* and *RM_VP* in these equations represent respectively Identified and Actual members of VP_i .

We randomly generate two sets from *ArgoUML-SPL*’s FM and three sets from *MobileMedia* using FeatureIDE tool. Each generated set has different size and it also covers all features shown in its corresponding FM. The *set1* in all case studies represents all possible configurations can be generated from its corresponding FM.

$$Precision(VP_i) = \frac{|IM_VP_i \cap RM_VP_i|}{|IM_VP_i|} \times 100\% \quad (1)$$

$$Recall(VP_i) = \frac{|IM_VP_i \cap RM_VP_i|}{|RM_VP_i|} \times 100\% \quad (2)$$

Table 2 shows obtained results by applying our algorithms to product configurations generated from FMs of case studies considered. In this table, we present Precision and Recall for each identified VP and average Precision and Recall for all identified VPs corresponding in each set of configurations. Highlighted rows refer to VPs that identified by our algorithms but they actually are not present in FMs of cases studies considered (false-positive VPs).

In *MobileMedia* case study, the proposed algorithms give 100% *Precision* and 100% *Recall* for each VP in both

²<http://www.ic.unicamp.br/~tizzei/mobilemedia/>

Table 2: Precision and Recall of Identified VPs of Features for Both ArgoUML-SPL and MobileMedia.

ArgoUML-SPL				
Set1: No. Configurations = 256 (All possible configurations)				
	Precision	Recall	Average Precision	Average Recall
Mandatory	100%	100%	58%	67%
OR-Group VP1	67%	100%		
OP-Group VP1	0.0%	0.0%		
Set2: No. Configurations = 7				
Mandatory	100%	100%	58%	67%
OR-Group VP1	67%	100%		
OP-Group VP1	0.0%	0.0%		
MobileMedia				
Set1: No. Configurations = 16 (All possible configurations)				
Mandatory	100%	100%	100%	100%
OR-Group VP1	100%	100%		
OP-Group VP1	100%	100%		
Set2: No. Configurations = 8				
Mandatory	100%	100%	25%	25%
OR-Group VP1	0.0%	0.0%		
OP-Group VP1	0.0%	0.0%		
XOR-Group VP1	0.0%	0.0%		
XOR-Group VP2	0.0%	0.0%		
Set3: No. Configurations = 5				
Mandatory	100%	100%	100%	100%
OR-Group VP1	100%	100%		
OP-Group VP1	100%	100%		

set1 (containing all possible configurations) and set3 (containing only 5 configurations). This is because these configurations have a high diversity of feature combinations to detect the behavior of members of each VP. In set 2, although the number of configurations is half of all possible configurations, the proposed algorithms fails to identify two VPs (*OR-Group VP1*, *OP-Group VP1*) and return two false positive VPs (*XOR-Group VP1*, *XOR-Group VP2*).

In ArgoUML-SPL, the identified VPs from set1 and set2 have the same *Precision* and *Recall* values in spite of set1 represents all possible configurations while set2 represents very small number of configurations. This is due to the fact that the FM of ArgoUML-SPL just offers two types of VPs (*OR-Group VP1*, *OP-Group VP1*) which this means that we only rely on feature descriptions to identify these VPs and do not pay attention to the number of available configurations. We also notice that the algorithms failed to identify *CognitiveSupport* and *Logging* (their label is *OP-Group VP1*) as VP of type *OP-Group* but they are identified as *OR-Group*. This is because these features share keywords with all other features as they have a crosscutting behavior in all ArgoUML-SPL features. Therefore, (*CognitiveSupport* and *Logging*) are identified as members of *OR-Group VP1*. This leads to degrade *Precision* and *Recall* values of *OP-Group VP1* and *OR-Group VP1* as shown in Table 2.

4.3 Validating the Identification of VPs of Components

For space limitation, we present only the result of applying our approach to ArgoUML-SPL case study. We generate 7 products corresponding to the second set of configurations in ArgoUML-SPL (see Table 2).

Table 3: Common Components and Variation Points at Architectural Level for ArgoUML-SPL.

At Architecture Level: OR-VP1		
At Feature Level: OR-Group VP1 (State, Activity, UseCase, Collaboration, Deployment, Sequence)		
Parent Components	Member Components	
Fig Prop Diagram Action Mode List State	Action Fig Button New Prop State Selection	
Fig Diagram State Sequence Model Activity Renderer	UML Model List Fig Prop Case Use	
UML Fig Diagram Init Model Action List	Action Fig Mode Iterator Message State Attributes	
Go Fig To State Diagram Collaboration Machine		
At Architecture Level: OP-VP1		
At Feature Level: OP-Group VP1 (CognitiveSupport, Logging)		
Parent Components	Member Components	
No Parent Components	Cr Wiz Child Many Conflict No Gen	
	Cr To Name Do By Missing Class	
	To Go Cr Init Wiz	
	Cr Resolved Abstract Transitions Name	
	Wiz Default Step To Renderer Tree Prop	
	Cr Goals Dialog Add Param Type To	
	Node Decision Knowledge Goal Priority Type	
	Cr Go Wiz No Name Operation Invalid	
	List Action To Object Tab Do	
	Table Cr Checklist To Abstract UML Model	
	At Architecture Level: Mandatory Components	
	At Feature level: Mandatory Features: Class	
	Parent Components	Member Components
No Parent Components	Fig Style List Panel Class diagram Model	
	Package Character Port Rect Fig	
	Action Show Hide Visibility Stereotype	
	Classdiagram Fig Subsystem Association Edge	
	Classdiagram Fig Event Edge Object	
	Selection Fig Class Node List Stereotype	
At Architecture Level: Shared-Com		
Cr Without Instance Classifier		
Cr Without Class Component		
Cr Without Instance Node Comp		
Cr Without Instance Classifier Node Collection Iterator		
Cr Without Node Component		
Cr Without Instance Classifier Component		
Cr Interface Without Component		

Table 3 shows the identified mandatory components and VPs of components by applying our approach to the 7 products of ArgoUML-SPL code base. *Parent Components* column presents components that form the parent of a VP while *Member Components* column presents components that form members of that VP. In this table, we show for each VP at the architecture level its corresponding VP at the feature level. The name of the identified VPs of components in this table as a follows: *OR-VP1* and *OP-VP1*. For components of *OR-VP1*, we notice that all names of parent components share the term “Diagram”. This means that they are not specific to certain feature (UML Diagram) but they may support all UML diagrams. By referring to ArgoUML-SPL’s FM, we can find out that the group of features corresponding to *OR-VP1* is under title “Diagrams”. Consequently, our approach can distinguish between parent components and member components. For components of *OP-VP1* which is corresponded to *CognitiveSupport* and *Logging* features, our approach identifies no parent components for this VP. This is because its components are only extracted from the implementation of (*CognitiveSupport*) while the *Logging* feature is implemented by external library (Log4J) [8]. From the names of these components, we can notice that they specific to *CognitiveSupport* because their names contain “Cr” term which refers to *Critics* supported by *CognitiveSupport* feature. For mandatory

components which are extracted from the implementation of mandatory feature (Class diagram), the names of these components show that these components implement this feature as their names contain the term “Classdiagram”. Of course, there is no parent components for mandatory components because there is only one feature. For components of *Shared-Com*, all components of this group related to only *CognitiveSupport* as their names include “Cr” term. This is expected because this feature has across cutting behavior through all other features. This means the implementation of this feature is shared among the implementation of other feature groups, which that represents the semantic of *Shared-Com* group. Consequently, our approach can extract and determine components that are shared between VPs of components (*Shared-Com*).

5 Related Work

The existing works support only forward engineering way for building SPLA. In [9], Sochos et al. propose to create SPLA based on FM. A strong mapping between features and components are established based on four transformations on the initial FM leading to SPLA. According to their approach components are developed from scratch to implement the transformed features. In [10], Trinidad et al. propose to automatically build a component model from a FM for developing dynamic SPL. They create for each feature a component and relations among features become relations among components. In [11], Zhang et al. propose to map feature to architectural components for building SPLA. In their approach, features are classified according to the variability type into *mandatory* and *optional*. In their approach, a component is created for each feature. The components of crosscutting features are implemented by object-oriented techniques while the components of non-crosscutting features are implemented by aspect-oriented techniques.

6 Conclusions

In this paper, we have proposed an approach for recovering SPLA from software product variants. Our approach focused on identifying mandatory components and variation points of components as an important step toward recovering SPLA. We analyzed commonality and variability across product variants in terms of features, as they represent the main source of commonality and variability in SPLA. In our experimental evaluation using two case studies, we showed that if we have small number of configurations with high diversity and precise feature descriptions, our approach achieves high precision and recall of identified variation points of features, and hence this leads to identify relevant variation point of components.

References

- [1] K. C. Kang, S. G. Cohen, J. A. Hess, W. E. Novak, and A. S. Peterson, “Feature-oriented domain analysis (foda) feasibility study,” November 1990.
- [2] F. J. v. d. Linden, K. Schmid, and E. Rommes, *Software Product Lines in Action: The Best Industrial Practice in Product Line Engineering*. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2007.
- [3] K. Pohl, G. Bckle, and F. J. van der Linden, “Software product line engineering: Foundations, principles and techniques.” Springer Publishing Company, Incorporated, 2010.
- [4] L. de Silva and D. Balasubramaniam, “Controlling software architecture erosion: A survey,” *J. Syst. Softw.*, vol. 85, no. 1, pp. 132–151, Jan. 2012.
- [5] S. Chardigny, A. Seriai, M. Oussalah, and D. Tamzalit, “Extraction of component-based architecture from object-oriented systems,” in *WICSA*, 2008, pp. 285–288.
- [6] D. Benavides, S. Segura, and A. Ruiz-Cortés, “Automated analysis of feature models 20 years later: A literature review,” *Inf. Syst.*, vol. 35, no. 6, pp. 615–636, Sep. 2010.
- [7] E. N. Haslinger, R. E. Lopez-Herrejon, and A. Egyed, “Reverse engineering feature models from programs’ feature sets,” ser. WCRE ’11. Washington, DC, USA: IEEE Computer Society, 2011, pp. 308–312.
- [8] M. V. Couto, M. T. Valente, and E. Figueiredo, “Extracting software product lines: A case study using conditional compilation,” ser. CSMR ’11. Washington, DC, USA: IEEE Computer Society, 2011, pp. 191–200.
- [9] P. Sochos, M. Riebisch, and I. Philippow, “The feature-architecture mapping (FArM) method for feature-oriented development of software product lines,” ser. ECBS ’06. Washington, DC, USA: IEEE Computer Society, 2006, pp. 308–318.
- [10] P. Trinidad, A. R. Corts, J. Pena, and D. Benavides, “Mapping feature models onto component models to build dynamic software product lines.” in *SPLC (2)*. Kindai Kagaku Sha Co. Ltd., Tokyo, Japan, 2007, pp. 51–56.
- [11] J. Zhang, X. Cai, and G. Liu, “Mapping features to architectural components in aspect-oriented software product lines,” ser. CSSE ’08. Washington, DC, USA: IEEE Computer Society, 2008, pp. 94–97.

Improving the Applicability of *Bayesian Networks* through Production Rules

Raissa da Silva^{*1}, Mirko Perkusich^{†1}, Renata Saraiva^{‡1}, Arthur Freire^{§1}, Hyggo Almeida^{¶1}, and Angelo Perkusich^{||1}

¹Embedded and Pervasive Computing Laboratory, Federal University of Campina Grande, Brazil

Abstract

One of the key challenges in constructing a Bayesian network BN is defining the node probability tables (NPT). For large-scale BN, learning NPT through domain experts knowledge elicitation is unfeasible. Previous works proposed solutions to this problem using the concept of ranked nodes; however, they have limited modeling capabilities or rely on BN experts to apply them, reducing their applicability. In this paper, we present an expert system based on production rules to define NPTs with the purpose of enabling the definition of NPTs by experts with no ranked nodes-specific knowledge. To create the rules, we elicited data from an expert in ranked nodes. To validate our approach, we executed an experiment with a BN already published in the literature to verify if, with our approach, a practitioner can achieve the same or better configuration for the NPTs. We used the Brier score to assess the NPTs accuracy and evaluated the results with the Wilcoxon test. All the Wilcoxon tests executed rejected the null hypotheses that stated that the Brier scores for the original NPTs method were the same as the new NPTs. By using our solution, a practitioner can accurately define NPTs without understanding the concept of ranked nodes.

Bayesian network, Expert systems, Production rules, Knowledge acquisition

^{*}raissa.silva@embedded.ufcg.edu.br

[†]mirko.perkusich@embedded.ufcg.edu.br

[‡]renata.saraiva@embedded.ufcg.edu.br

[§]arthur.freire@embedded.ufcg.edu.br

[¶]hyggo@embedded.ufcg.edu.br

^{||}perkusic@embedded.ufcg.edu.br

1. Introduction

BNs are probabilistic graph models and are used to represent knowledge about an uncertain domain [1]. BNs have been applied to develop expert systems for many contexts such as software process management [12] and effort estimation of web development projects [8]. There are two challenges to build a BN: building the directed acyclic graph (DAG) and defining the NPTs. In this paper, we focus on the second challenge.

A BN's NPT can be automatically learnt from data [6] or by domain expert elicitation [4]. In practice, it is rare to have an adequate database [4] and it becomes necessary to elicit data from domain experts. However, manually defining the NPTs through domain experts can become unfeasible depending on the number of nodes and states, because the complexity grows exponentially.

To reduce the effort of defining NPTs through domain experts, Fenton et al. [4] proposed the concept of ranked nodes. It consists of eliciting data from the expert through a truth table composed of ordinal elements using four types of weighted functions. Given the collected data, the calibration of the NPT (i.e., type of function, weights and variance) is defined. The authors applied the approach in two cases: safety assessment and software defect prediction, saving 84% and 93%, respectively, of effort compared to a manual approach. On the other hand, to use this solution it is necessary to understand the concept of ranked nodes, because given the data collected from the expert, it is necessary to manually calibrate the NPTs. Therefore, it is unpracticable for domain experts with no ranked nodes-specific knowledge.

Perkusich et al. [11] presented an approach based on the concept of ranked nodes. They focused on encapsulating *Bayesian networks*-specific knowledge from the practitioner and reducing the effort to collect data from the expert. On the other hand, it is limited to only one of the four functions presented by Fenton et al. [4]. Therefore, it has

limited modeling capabilities.

Our goal is to combine the strengths of the solutions presented in Fenton et al. [4] and Perkusich et al. [11]. With this purpose, we automatized the approach presented in Fenton et al. [4] by using *production rules*. To create the rules, we elicited data from a BN expert to, for a set of combination of evidences of the parent nodes, define the best configuration of the NPT.

To validate our approach, we executed an experiment and used a *Bayesian network* already published in the literature as the object of study. We randomly selected five nodes from the given *Bayesian network* as the objects of study and used them to verify if, with our approach, a practitioner can achieve the same or better configuration for the NPTs. For each node, we randomly selected twelve combinations of states to elicit data from one practitioner, calculated the Brier score and evaluated the results with the Wilcoxon test. All the Wilcoxon tests executed rejected the null hypotheses that stated that the Brier score for the old method was the same as the new. Therefore, we conclude that we considerably improved the accuracy of the model presented in Perkusich et al. [10].

This paper is organized as follows. Section 2 presents background on *Bayesian networks* and ranked nodes. Section 3 presents the methodology used to build and evaluate our solution. Section 4 presents the limitations of the solution and threats to validity. Section 5 presents our conclusions and future work.

2. Background

Bayesian networks are probabilistic graph models and are used to represent knowledge about an uncertain domain [1]. A *Bayesian network*, B , is a directed acyclic graph that represents a joint probability distribution over a set of random variables V [5]. The network is defined by the pair $B = \{G, \Theta\}$. G is the directed acyclic graph in which the nodes X_1, \dots, X_n represent random variables and the arcs represent the direct dependencies between these variables. Θ represents the set of the probability functions. This set contains the parameter $\theta_{x_i|\pi_i} = P_B(x_i|\pi_i)$ for each x_i in X_i conditioned by π_i , the set of the parameters of X_i in G . Equation 1 presents the joint distribution defined by B over V .

$$P_B(X_1, \dots, X_n) = \prod_{i=1}^n P_B(x_i|\pi_i) = \prod_{i=1}^n \theta_{X_i|\pi_i} \quad (1)$$

There are two challenges to build *Bayesian networks*: building the directed acyclic graph (DAG) and defining the NPTs [9]. In this paper, we focus only on defining the NPTs. For this purpose, there are two techniques: through (i) databases and (ii) domain experts [11]. Defining probability functions from databases can be automated by a pro-

cess called *batch learning* [6]. However, for many practical problems one rarely finds an adequate database. On the other hand, manually defining probability functions through domain experts can become unfeasible depending on the number of nodes and states. As shown in the work of Fenton et al. [4], inconsistencies could occur if domain experts try to elicit exhaustively the probability function for a node with a large number (e.g., 125) of states.

There are several methods to reduce this complexity and to encode expertise in large scale probability functions. Fenton et al. [4] proposes an approach for *Bayesian networks* composed of ranked nodes, which are the only types of nodes used in Perkusich et al. [10]. Ranked nodes have an ordinal scale (e.g., [Low, Medium, High]) and are based on the doubly truncated Normal distribution (TNormal) limited in the $[0, 1]$ region. This distribution is based on four parameters: μ , mean (i.e., central tendency); σ^2 , variance (i.e., confidence in the results); a , lower bound (i.e., 0); and, b , upper bound (i.e., 1). This distribution enables us to model a variety of shapes (i.e., relationships) such as a uniform distribution, achieved when $\sigma^2 = \infty$, and highly skewed distributions, achieved when $\sigma^2 = 0$.

In the approach presented in Fenton et al. [4], μ is defined by a weighted function of the parent nodes. There are four weighted functions: weighted mean (WMEAN), weighted minimum (WMIN), weighted maximum (WMAX) and mixture of WMIN and WMAX (MIXMINMAX). According to the authors, these functions are enough to represent the types of relationship necessary for defining the probability function.

To define which function should be used, the model developer must perform “what if” analysis with the expert by defining questions and collecting answers to define a truth table. The model developer must analyze the answers and define the most appropriate function. The variance is defined empirically and it should reflect the expert’s confidence in the results [4]. We show an example of questions and answers in Table 1, in which the node C has two parents, A and B . In this example, since C tends to be equal to the smallest value of its parent nodes, the most appropriate function is WMIN.

Table 1. Example of truth table to define the weighted function for μ .

A	B	C
Very high	Very high	Very high
Very low	Very low	Very low
Very low	Very high	Very low
Very high	Very low	Low

Perkusich et al. [10] presented a simplified approach to define the probability functions based on the one presented

by Fenton et al. [4]. Instead of “what if” analysis, it orders the relationships between the child and parents nodes given their relative magnitude. With this purpose, it uses a questionnaire to elicit knowledge from experts. For each child node in the model, there is a question in the questionnaire. The questions are based on a template. The collected data is analyzed using statistical methods and used as input to an algorithm, which is presented in Perkusich et al. [10], that defines the weights for the function of μ . The advantage of this approach is its simplicity to elicit knowledge from several experts and encapsulation of ranked node and *Bayesian networks*-specific knowledge. The disadvantage is its modeling limitation due to only using one type of function (WMEAN) and fixed variance of $5.0E^{-4}$. Furthermore, given results of a case study presented in Perkusich et al. [12], this approach is too abstract and sensible to errors.

3. Methodology

Our main goal is to increase the applicability of using BN by encapsulating the complexity of calibrating the NPTs from the domain experts. With this purpose, we present an expert system based on *production rules* to, given a set of input values, automatically calibrate a NPT. Our solution is based on the concept of ranked nodes. Therefore, with our solution, domain experts can calibrate the NPTs without the need to understand how ranked nodes work. We evaluated our solution with an experiment using a BN already published in the literature. In Section 3.1, we define the problem in details. In Section 3.2, we present details about our solution. In Section 3.3, we present the process and results of our empirical evaluation.

3.1. Problem definition

More specifically, our goal is to combine the modeling capabilities of the approach presented in Fenton et al. [4] and the ranked nodes-specific knowledge encapsulation of the approach presented in Perkusich et al. [11]. To elicit knowledge from experts, as in Fenton et al. [4], we use “what if” analysis (i.e., truth table results). Given the information collected, we automate the calibration of the probability function.

To calibrate the probability function of a ranked node it is necessary to define three parameters: f , $V = (v_1, \dots, v_k)$ and σ^2 , where f is the type of function, V is a vector containing parent node’s weights and k is the number of parent nodes. In AgenaRisk¹, these variables have the following range: $f \in \{WMEAN, WMIN, WMAX, MIXMINMAX\}$, $w \in \{1, \dots, 5\}$, $\sigma^2 \in \{5.0E^{-4}, \dots, \infty\}$ and $k \in \{1, \dots, \infty\}$.

Given that a probability function’s parameters are defined, we can assess the calculations’ (i.e., predictions) ac-

curacy with the Brier score [2]. For a single prediction, which is our case, it is simply the square of the difference between the predicted probability (q) and the actual outcome (o) [3], for each state: $B = \sum_{n=1}^s (o_n - q_n)^2$, where B is the Brier score and s is the number of possible outcomes (i.e., number of states of the given node). Given that we want the best possible calibration, the problem is to, given data collected from the experts, find a combination of parameters f and $V = (v_1, \dots, v_n)$ that minimizes B .

3.2. Solution

Our solution is an expert (i.e., production) system to emulate the knowledge of a specialist on ranked nodes. With this purpose we used *production rules*. A *production rule* consists of two parts: a sensory precondition (i.e., IF statement) and an action (i.e., THEN). If an input to the system matches a precondition, an action is triggered. With *production rules* it is possible to represent an expert knowledge. For instance, the given rule represents our knowledge regarding traffic: “if the traffic light is red then stop”. To define the rules, we relied on the knowledge of an expert with five years of experience using ranked nodes. We implemented the solution in Expert Sinta².

In Fenton et al. [4], the authors present the usage of a truth table composed of a combination of states of the parent nodes to collect data from domain experts. Therefore, the first step of our solution was to define which values the truth table should have. With the truth table (i.e., combinations) defined, we could define the preconditions of the system.

To elicit the weights for WMAX and WMEAN, Laitila [7] recommends that the expert specifies to which point (i.e., state) the mode of the child node rise when $s_a = (0)_{i=1}^n$ changes into $s_b = (0, \dots, 0, s_k = 1, 0, \dots, 0)$; and, for WMIN, the mode of the child node drops when $s_a = (1)_{i=1}^n$ changes into $s_b = (1, \dots, 1, s_k = 0, 1, \dots, 1)$. Therefore, for each configuration (i.e., number of parent nodes), we considered $2 * n$ cases, in which n is the number of parent nodes. The combinations used to calibrate a child node with three parents (A, B and C) considering that all nodes are composed of the states $s = (Verylow, Low, Medium, High, VeryHigh)$ is shown in Table 2. For $n = 2$, we added the combinations $(VeryLow, Medium)$ and $(Medium, VeryLow)$.

Then, the expert defined, for each possible combination in the truth table, the best calibration for the NPT (i.e., the action): function type and weights. We defined $\sigma^2 = 5.0E^{-4}$, because, according to the expert, changing the functions and weights is enough. In fact, in Perkusich et al. [10], the authors defined $\sigma^2 = 5.0E^{-4}$ and had success. For instance, if for the combination $(VeryHigh, VeryLow)$ the expected value is *Low*; for $(VeryLow, VeryHigh)$, is *Low*; for $(VeryLow, Medium)$, is *Verylow*; and for

¹www.agenarisk.com

²<http://www.lia.ufc.br/bezerra/exsinta/>

Table 2. Truth table for a child node with three parent nodes.

A	B	C
Very high	Very high	Very low
Very high	Very low	Very high
Very low	Very high	Very high
Very low	Very low	Very high
Very low	Very high	Very low
Very high	Very low	Very low

(*VeryHigh, VeryLow*), is *Verylow*, then the best calibration is: WMIN function, with weights 3 and 5. To verify the calibrations, we used AgenaRisk and the Brier score. To consolidate a rule, the mean Brier score for all combinations was lower than 0.1.

The files with the rules defined by the expert are available in a website³. We only defined rules for child nodes with two and three parents, because whenever a node has more than three parents, divorcing should be used to simplify the BN [3].

3.3. Empirical evaluation

To evaluate our solution, we executed an experiment by using a sample of nodes of the BN presented in Perkusich et al. [10] as the object of study. The given BN was chosen due to availability and it models the key factors of *Scrum*-based software projects with the goal of assisting on the continuous improvement of the team and processes. The BN is composed of twenty child nodes. In other words, there are twenty NPTs to be calibrated. The NPTs were calibrated using the approach presented in Perkusich et al. [11] by collecting data from forty practitioners.

The evaluation consists of applying our solution to calibrate a sample of NPTs from the object of study and compare the accuracy of the new NPTs and the old ones. For this purpose, we elicited knowledge from one expert (i.e., subject), whom has five years of experience working on *Scrum* projects as a *Scrum Master* and was already familiarized with the BN presented in Perkusich et al. [10]. First, we elicited knowledge regarding the expected value of the NPT for a set of combinations of the states of the parent nodes to use as input to our system, I . Afterwards, we elicited knowledge for a different set, E , in which $E \cap I = \emptyset$, and compared it with the system's calculated values. In this case, since we compare data generated by our solution with the subject's, there is no conflict on having the same subject to calibrate and evaluate the NPTs. We focused on the following research question and informal null hypothesis:

³<https://seke2016.wordpress.com/expertsinta-files/>

RQ1: Comparing with the old model, does using the new calibration maintain or improve the model's accuracy given the expert's expectation?

H₀: Accuracy decreases.

Due to space limitation, we only present data used to calibrate two nodes. The data collected for all nodes is presented in a website⁴. In Figure 1, we present a summarized view of the BN, in which the boxes represent a set of nodes in the original BN.

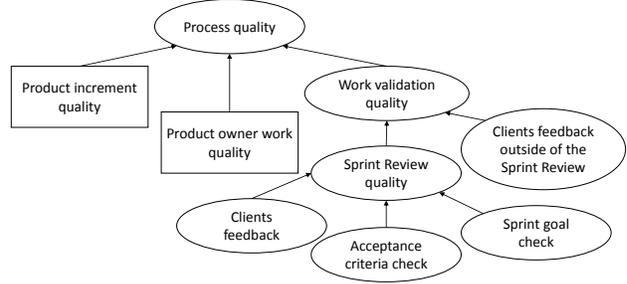


Figure 1. Summarized view of the BN used as object of study.

In Table 3, we present the data collected to elicit the table for the node *Work validation quality*. For this node, by using the rules defined in our system, the calculated calibration was: $f = WMEAN$ and $V = (2, 1)$. In Table 4, we present the data collected to elicit the table for the node *Sprint Review quality*. For this node, the calculated calibration was: $f = WMIN$ and $V = (3, 3, 3)$.

For the experiment, we randomly selected five child nodes, with two or three parents, of the BN presented in Perkusich et al. [10] as the objects of study. The response variables are the old and new models' accuracy, which we are assessed by the Brier scores. For each node, we randomly defined twelve combinations of parent nodes' states and used them to, through a truth table, elicit data from the expert regarding the expected central tendency of the given node. For each combination, we calculate the Brier score using the calibration presented in Perkusich et al. [10] and using our calibration. We used the average Brier score to compare the models' accuracy. Given that the data did not follow a Normal distribution, we used the Wilcoxon test. By analyzing the results of the Wilcoxon tests, one for each node, we assessed **RQ1**.

The objects of the study were the nodes: *Work validation quality*, *Product backlog quality*, *Software engineering techniques quality*, *Sprint Review quality* and *Product Backlog is properly ordered*. Due to space limitation, in Table 5, we only show the elicited data and calculated Brier scores for the node *Work validation quality*. For the *old* (i.e., Perkusich et al.'s [10]) model, the average Brier score

⁴<https://seke2016.wordpress.com/2016/03/17/seke-2016/>

Table 3. Data elicited for the node *Work validation quality*.

Sprint Review quality	Clients feedback outside of the Sprint Review	Work validation quality
Very low	Very high	Low
Very high	Very low	High
Very low	Medium	Low
Medium	Very low	Low

Table 4. Data elicited for the node *Sprint Review quality*.

Stakeholder feedback	Sprint goal check	Acceptance criteria check	Sprint Review quality
Very low	Very high	Very low	Low
Very high	Very low	Very low	Low
Very low	Very low	Very high	Low
Very low	Very high	Very high	High
Very high	Very low	Very high	Low
Very high	Very high	Very low	Medium

is 0.59 with $\sigma = 0.61$. For the *new* (i.e., our) model, the average Brier score is 0.24 with $\sigma = 0.35$. By applying the Wilcoxon test with $\alpha = 0.5$, we had $p - value = 0.0042$. Therefore, we reject the null hypothesis that states that the median of the Brier score for the NPTs defined with our approach are worse than the original.

For the node *Product backlog quality*, we had $p - value = 0.0085$. For *Software engineering techniques quality*, $p - value = 0.041$. For *Sprint Review quality*, $p - value = 0.033$. For *Product Backlog is properly ordered*, $p - value = 0.0017$. Therefore, for all nodes, we conclude that the new model is more accurate. A threat to validity is that we might not have evaluated enough nodes to assess **RQ1**.

4. Limitations

The limitations of this study are related to the production rules definition and threats to validity regarding the experiment. Regarding the production rules definition, we only relied on the experience of one expert to define them. To minimize this limitation, we selected an expert experienced with ranked nodes and we used the Brier score to minimize the chances of an incorrect rule definition. Furthermore, the proposed system only handles child nodes with two or three parents. However, in practice, this should not limit its application, because whenever a node has more than three parents, divorcing should be used to simplify the BN [3]. Additionally, the solutions were defined for ranked nodes composed of a 5-point Likert scale. Finally, the definition of the rules are based on values of AgenaRisk. On the other hand, currently, it is the only tool that implements ranked nodes.

Regarding the limitations of the experiment, it has conclusion, internal, and external threats to validity. The con-

clusion threats to validity are related to the sample sizes used for the objects of study. The original BN was composed of twenty child nodes and we only evaluated five. Furthermore, to compare the accuracies, we only evaluated twelve combination of states. The internal threats to validity are related to the subject selection process. On the other hand, we minimized this threat by choosing an expert familiarized with the BN, which minimized the threat of eliciting inconsistent knowledge. The external threats to validity concern the ability to generalize experiment results outside the experiment setting. Since we only one BN and one subject, we cannot generalize our results. However, given that ranked nodes are used and the data collected from the domain expert is consistent, there is no reason to believe that our system would not output accurate data.

5. Conclusion

In this paper, we presented an expert system to, given knowledge elicited from the domain expert, automate the definition of NPTs of BN. Our solution is based on ranked nodes and decreases the complexity of defining NPTs. Furthermore, it increases the applicability of using BN, because it encapsulates from the domain expert the complexity regarding calibrating the NPTs.

We improved the method of defining the probability functions of the model presented in Perkusich et al. [10] by automating the approach presented in Fenton et al. [4] using *production rules*. We used the method to elicit data from one expert and calibrate the model.

To evaluate our solution, we executed an experiment with five randomly selected nodes, with two or three parents, from a BN presented in the literature [10] as the objects of the study, in which we compared the accuracy of NPTs defined with our approach with the originals. For

Table 5. Data collected and calculated Brier scores for the node *Work validation quality*.

Sprint Review quality	Clients feedback outside of the Sprint Review	Work validation quality	Old Brier score	New Brier score
Very low	High	Low	0.5	0.0085
Low	High	Low	1.8	0.97
High	Low	High	1.8	0.97
Low	Medium	Low	0.5	0.18
Medium	Low	Medium	0.5	0.18
Medium	Very high	High	0.0041	0.18
Very high	Medium	High	0.0041	0.18
Very high	Low	High	0.5	0.0077
High	Very low	Medium	0.5	0.0077
Low	Very high	Medium	0.5	0.0077
Medium	High	Medium	0.5	0.18
Very low	Very low	Very low	0.0021	0.0013

each node, we randomly selected twelve combinations of states to elicit data from the expert, calculated the Brier score and evaluated the results with the Wilcoxon test. All the Wilcoxon tests executed rejected the null hypotheses that stated that the Brier score for the old method was worse than the new. Therefore, we concluded that, with our solution, a domain expert can calibrate NPTs with the same accuracy as with Fenton et al. [4], because the expert system was built following their approach, and without ranked nodes-specific knowledge.

For future works, we intend to investigate the risks on using ordinal scales to elicit expert knowledge and use fuzzy logic to model the elicited data from experts. Furthermore, we plan on developing a tool that supports ranked nodes and use genetic algorithms to calibrate the probability functions.

References

- [1] I. Ben-Gal. *Bayesian Networks*. John Wiley and Sons, 2007.
- [2] G. W. Brier. Verification of forecasts expressed in terms of probability. *Monthly Weather Review*, 78(1):1–3, 1950.
- [3] N. Fenton and M. Neil. *Risk Assessment and Decision Analysis with Bayesian Networks*. CRC Press, 5 edition, 11 2012.
- [4] N. E. Fenton, M. Neil, and J. G. Caballero. Using ranked nodes to model qualitative judgments in bayesian networks. *IEEE Trans. on Knowl. and Data Eng.*, 19(10):1420–1432, Oct. 2007.
- [5] N. Friedman, D. Geiger, and M. Goldszmidt. Bayesian network classifiers. *Machine Learning*, 29(2-3):131–163, 1997.
- [6] D. Heckerman. Learning in graphical models. chapter A Tutorial on Learning with Bayesian Networks, pages 301–354. MIT Press, Cambridge, MA, USA, 1999.
- [7] P. Laitila. Improving the Use of Ranked Nodes in the Elicitation of Conditional Probabilities for Bayesian Networks. Master’s thesis, Aalto University, Espoo, Finland, 2013.
- [8] E. Mendes. Using knowledge elicitation to improve web effort estimation: Lessons from six industrial case studies. In *Software Engineering (ICSE), 2012 34th International Conference on*, pages 1112–1121, June 2012.
- [9] M. Neil, N. Fenton, and L. Nielson. Building large-scale bayesian networks. *Knowl. Eng. Rev.*, 15(3):257–284, Sept. 2000.
- [10] M. Perkusich, H. O. de Almeida, and A. Perkusich. A model to detect problems on scrum-based software development projects. In *Proceedings of the 28th Annual ACM Symposium on Applied Computing, SAC ’13*, pages 1037–1042, New York, NY, USA, 2013. ACM.
- [11] M. Perkusich, A. Perkusich, and H. Almeida. Using survey and weighted functions to generate node probability tables for *Bayesian* networks. In *Proceedings of BRICS-CCI 2013*, 2013.
- [12] M. Perkusich, G. Soares, H. Almeida, and A. Perkusich. A procedure to detect problems of processes in software development projects using bayesian networks. *Expert Systems with Applications*, 42(1):437–450, 2015.

Component-based process

For modeling language evaluation

Khaoula Sayeb, Oualid Khayati, Naoufel Kraeim

RIAD-ENSI

University of Manouba

Manouba, Tunisia

firstname.lastname@gmail.com

Abstract—This paper presents a component-based approach to build a modeling language evaluation process. We firstly define a process components repository that capitalizes and implements solutions for modeling language evaluation. We describe the process component model. Then we describe how we reuse the process components repository to define a process for modeling language evaluation based on pre-built components.

Keywords—process component; modeling language; evaluation; component model; reuse; components repository.

I. INTRODUCTION

One of the major evolutions in software development is the component-oriented programming approach. It facilitates the construction of complex applications, their deployment, their administration and their evolution control. The rise of software components produces two types of complementary processes:

- The design for reuse: components engineering needed to create, enrich and maintain a repository of reusable components. It implements the identifying features, specification, development, validation and organization of components.
- The design by reuse: components based engineering consists on the reuse of pre-built components to define new products.

Components-based approach is applied to software engineering, method engineering and also process engineering. In our approach, we propose a process components repository that we use later to define an evaluation process. Each process component describes a modeling language (ML) evaluation solution. It can be then composed and connected with other components to build a ML evaluation process. The aim of this paper is to present the process component model. This later describes process components structure, relationships and composition rules. We present the process components repository and how we reuse it to define a component-based process for ML evaluation.

This paper is structured as following: In the second section, we give an overview about ML evaluation. We introduce our process component model in section three. Section four is dedicated to describe the components-based process for ML evaluation and to give a demonstration example that explains our approach. We finish this paper by the conclusion.

II. MODELING LANGUAGE EVALUATION

A ML has a key role in software modeling process. It affects directly the quality of software design (models) and so the final software quality. A ML is defined as the models expression language [1]. It is determined by a semantic (that is a set of concepts and rules that specify the field), a concrete syntax (which is a set of symbols that represent concepts) and an abstract syntax (expressed by meta-model) [2].

Although the ML evaluation domain is relatively recent, there are a huge variety of approaches and frameworks in this field. They propose solutions, instructions and features to qualify or quantify the ML quality. We propose to unify and organize terms on this domain. Then, each quality approach is called quality framework. For instance, the physics of notations (Moody's framework [3]) is a framework for concrete syntax evaluation, sequal (krogstie's framework [4]) proposes a generic framework for the whole language evaluation, etc. A quality framework is composed of a set of quality attributes. The concept of quality attributes unifies existing concepts in the literature such as dimensions, attributes, features or sub-features, criteria, factors, etc. The physics of notations proposes nine quality attributes to define the cognitive effectiveness of a visual notations in general and specifically graphical concrete syntax. Perceptual discriminability is a quality attribute that determines the ease and accuracy with which graphical symbols can be differentiated from each other. It uses other quality attributes to decompose the solution (perceptual discriminability uses visual distance and perceptual popout). To measure quality attributes, we use evaluation techniques that can be a metric or a qualitative or quantitative protocol. In addition, an evaluation technique offers a concrete outcome that estimates a quality attribute.

We use process components to model knowledge in the ML evaluation domain. Our first purpose is to provide a structured documentation about this field. Secondly, we aim to provide tools for the implementation and the built of a components-based process. Process components are defined following a model that we describe in the next section.

III. DESIGN FOR REUSE: PROCESS COMPONENT

In our approach, a process component is a component that provides a solution for ML. In this section, we present the process component model. A component model consists of a set of conventions to be followed in the construction and use of

components. It has to define the component structure, relationships and component reuse techniques.

A process component can be a conceptual component (a design pattern that describes a framework, a quality attribute or an evaluation technique) or a software component (that implements a conceptual component). A design pattern describes the context of the framework, the solution provided and the problem resolved by the framework. In addition, we use design pattern to capitalize knowledge about ML evaluation. A software component implements the solution offered by conceptual component. For instance, a conceptual component describes the solution to assess the visual distance between concrete syntax symbols. An associated software component takes in entry the list of concrete syntax symbols and measures the visual distance between them. The result of the conceptual component is a solution approach. The result of the software component is a significant value that represent the visual distance. The next subsections detail conceptual and software component models.

A. Conceptual component model

We use design pattern to describe our conceptual component. A design pattern is defined as a solution of a recurring problem in a context. The design pattern model specifies the structure adopted by the designer to represent patterns. It is composed of a set of rubrics. To define our process design pattern, we use the P-SIGMA [5] model that we adapt to take into account the capitalization needs. Moreover, we add some rubrics and customize others. P-Sigma is composed of three parts: Interface, Realization and Relation. **Interface** part contains all elements allowing pattern's selection. **Realization** part gives the pattern solution. Finally, the **relationship** part describe links between patterns. Fig.1 describes our adapted version of the P-SIGMA formalism (Adapted rubrics are gray). We add the *reference* rubric which gives the source of the approach described by the conceptual component. The *realize* rubric is added to define a new relationship. The *classification* rubric is customized to deal with our classification approach.

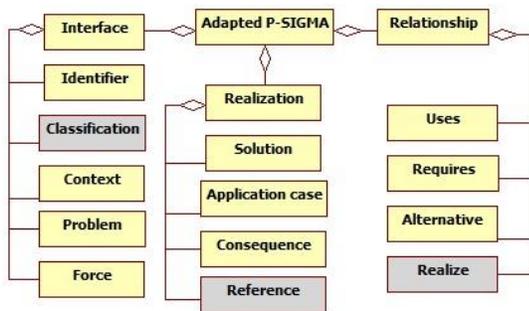


Figure 1. Adapted P-SIGMA

Conceptual components are described in more details with examples in a further work [6]. The following table (TABLE1) presents the conceptual component that describes the quality attribute: **perceptual discriminability**.

TABLE 1. CONCEPTUAL COMPONENT: PERCEPTUAL DISCRIMINABILITY

Interface
Identifier: perceptual discriminability

Classification: Graphical concrete syntax, semiotic framework, evaluation.
Problem: Are symbols distinguishable between each other? How to define ML graphic elements that are perceptually different?
Context: The construction of a new ML. The evaluation of the graphical concrete syntax.
Realization
Solution: we describe how to determine the perceptual discriminability [1]. It is too long to express here.
References: [1] D.L. Moody. The 'physics' of notations: Toward a scientific basis for constructing visual notations in software engineering. IEEE Transactions on Software Engineering, 2009.
Relationship
Use: Visual distance, perceptual popout.

B. Software component Model

The widely accepted definition of software components is that of [7]: "A software component is a unit of composition with contractually specified interfaces and explicit context dependencies only. A software component can be deployed independently and is subject to composition by third parties". A software component model is a definition of the semantics of components (that is, what components are meant to be), the syntax of components (that is, how they are defined, constructed, and represented), and the composition rules of components (that is, how they are composed or assembled) [8].

In this subsection, we present our software components model for ML evaluation. This model defines software components structure, relationships and its composition rules.

1) Software component structure

The component structure is presented in Fig.2. Depending of the component's architecture, we distinguish two component's types. *The primitive component* is a "basic" component. *The composite component* is an aggregation of several primitive or composite components.

Each component is a part of an evaluation process. It has to realize a processing that measures quality (a solution). It requires, for this aim, parameter of the evaluation and then it provides results. Therefore, a component is composed of a solution and a set of ports. The ports realize provided and required interfaces either to acquire parameters necessary for the assessment or to provide descriptions and results of the evaluation. Then a component can have a required interface "Evaluation parameters" for acquiring the necessary parameters of the evaluation and a "Primary result" interface for gathering primary results needed to calculate the result of the concerned component. "Primary result" interface is not mandatory for all components. For provided interface, a component must have *An evaluation purpose* and a *Final result* interfaces. The first one to describe the purpose for which this component is used and integrated into an evaluation process. The second one to provide the evaluation result. In the case of a primitive component, it will be a result of the evaluation. In the case of a composite component, this will be a combination of other components results. The combination is done using a formula defined in the component solution.

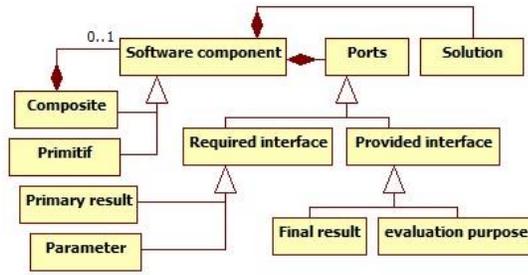


Figure 2. Software component structure

In this subsection, we defined the software components structure. The next step is the definition of relationships.

2) Relationship

As mentioned below, a software component implements a quality framework, a validation, a quality attribute or an evaluation technique. We have to define relationships between components. Furthermore, we have four types of relationships.

The composition relationship: one component is composed of other components. Therefore, its treatment is delegated to its composite components. Specifically in our approach, quality framework components and validation components must be composed of at least one quality attribute component.

The realization relationship: it connects two components where the solution of the former is more refined than that of the second. This relationship connects a quality attribute component and an evaluation technique component. Indeed an evaluation technique component realizes a quality attribute component by providing the tools to calculate it and enhance its solution.

The use relationship: it connects two quality attributes components. It allows the decomposition of a problem described by a component more elementary components. More specifically, in our case, a quality attribute component uses the results provided by the used quality attribute components.

The alternative relationship: it connects two evaluation techniques components that provide two alternative solutions for the same problem. Therefore, these components realize the same quality attribute component.

3) Composition rules

Composition specifies how components are interconnected. Compositions declare instances number of components and define their configuration. Furthermore, a composition specifies how the ports of those instances are wired, i.e., which connector is used for connecting which ports. In our approach, we define the following composition rules:

Hierarchical composition and encapsulation (built components, sub-components): Composite components (quality frameworks component and validation components) encapsulate all the components involved in their achievements. In this case, each port of the composite component should be linked to one or more interfaces of its son components. Especially, the final result of a composite component is calculated based on primary results of its sub-components.

Interconnection components throw connectors: In fact, the connector can assemble components using their provided and required interfaces. In our approach, we have two

interconnections throw connectors. The first case (Case1) when a quality attribute component uses other quality attribute components (in Fig.4 perceptual discriminability uses visual distance and perceptual popout). The second case (Case2) when evaluation technique components realize a quality attribute component (in Fig.4 metric for visual distance realizes visual distance. In both cases (Case1 and Case2), the connection is made between a required interface final result and one or more provided interfaces primary result.

A demonstration of composition rules is shown in section four where we present a minimal example of application of process component to define a process for ML evaluation.

IV. DESIGN BY REUSE: COMPONENT BASED PROCESS

Component-based software engineering aims to improve the software engineering process by providing reusable components. Following the process component model described in section three, we create a components repository that capitalizes knowledge about ML evaluation. Software components serve to build a component-based process. In addition, a ML assessor selects software components from the repository and implements an evaluation process. In this section, we firstly present an evaluation process model. Then we give a minimal example that instantiates it.

A. Evaluation process model

We propose a model for the ML evaluation process that resumes all related features (Fig.3). In addition, a ML evaluation process is applied to a subject which is the parameter of the evaluation and produces as result an execution report. It depends on the context and the needs of the assessor. The context may be a comparative study of existing MLs or an improvement and validation of a ML under construction. The subject of an evaluation process may be a ML [4], a part of the ML (i.e. concrete syntax [3]), a ML family (i.e. Business Process ML [9]) or even just a ML property (i.e. usability [10]).

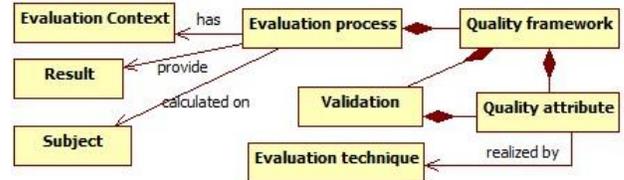


Figure 3. Evaluation process model

Besides, a ML evaluation process is composed of quality frameworks that provide solution for evaluating a ML. A quality framework is composed of validation and a set of quality attributes. A validation is an optional part in a quality framework. It provides an assessment of a ML by validating one of its parts relative to another as a set of quality attributes (for example evaluate the concrete syntax with respect to the abstract syntax). A quality attribute is realized by means of evaluation techniques (which calculate it throw metrics or throw protocols offering a concrete outcome of the evaluation).

A ML evaluation process is composed of one or many frameworks. A ML assessor builds an evaluation process by the selection of existing frameworks. We use the concept of process component to model these frameworks.

B. Evaluation process example

We propose a component base composed of conceptual and software components. Conceptual component capitalizes solution for ML evaluation. A ML assessor documents about the domain throw conceptual components. If he decides to implement an evaluation process, he has to define its context and its subject. Then he selects software components that compose the evaluation process. Its execution produces a report that resumes its application result. In this section, we give an evaluation process example that explains our solution and instantiates the proposed evaluation process model.

The process example:

Context: the evaluation of a graphical ML concrete syntax and its validation compared to the abstract syntax.

Subject: a graphical concrete syntax and an abstract syntax.

The evaluation process: the component diagram in Fig.4 represents software components that compose the example process. We use the framework proposed by Moody [4] for assessing graphical concrete syntax. It is composed of eight quality attributes to evaluate the cognitive effectiveness of a graphical concrete syntax and a quality attribute (semiotic clarity) that validates the concrete syntax with respect to the abstract syntax. In this example process, we just implement two of them (semiotic clarity and perceptual Discriminability).

Some informations are not represented to simplify the diagram. For instance, we had to wire each provided interface evaluation purpose (EP) of a composite component to the relative composed component. It is similar for the required interface parameter (P).

Abbreviation meaning in the Fig.4 are as following:

P: Parameter; **PR:** Primary result; **FR:** Final result; **EP:** Evaluation purpose.

Result: a report that gathers all final results in the order of the process execution and its composition to get the process result.

This process example have to be calculated on a graphical concrete syntax (i.e., that of UML) to acquire concrete result.

V. CONCLUSION

In this paper, we have presented a process component model that describes the process component structure, relationships and composition rules. In addition, we use two components types: conceptual components that describe a ML evaluation approach; and software components that implement it. Conceptual components provide a structured documentation. Software components are used to build a ML evaluation process. Benefits of using component to represent our process is that 1) we favor the capitalization and the reuse of ML evaluation works; 2) we build flexible process adapted to the assessment context and needs. We also propose an evaluation process model that describes ML evaluation process.

REFERENCES

- [1] Object Management Group, "Meta Object Facility (MOF) 2.0 Core Specification," 2006.
- [2] A. Kleppe, "A Language Description is More than a Metamodel," *ATEM*, 2007.
- [3] D. L. Moody, "The "Physics" of Notations: Toward a Scientific Basis for Constructing Visual Notations in software engineering," 2009.
- [4] J. Krogstie, "Evaluating UML using a generic quality framework," chez *UML and the unified process*, USA, IGI Publishing, Hershey, PA., 2003, pp. 1-22.
- [5] A. Conte, J.-P. Giraudin, J.-C. Freire Junior, I. Hassine and D. Rieu, "A tool and a formalism to design and apply patterns," *SugarloafPLOP*, 2002.
- [6] K. Sayeb, D. Rieu, S. Dupuy-Chessa et N. Mandran, "Qualité des langages de modélisation et des modèles: vers un catalogue des patrons collaboratifs," *INFORSID*, 2012.
- [7] C. Szyperski, "Component Software: Beyond Object-Oriented Programming", 2nd Addison-Wesley, 2002.
- [8] K.-K. Lau et Z. Wang, "A survey of software component models," *IEEE TRANSACTIONS ON SOFTWARE ENGINEERING*, 2007.
- [9] B. List et B. Korherr, "An Evaluation of Conceptual Business Process Modelling Languages," *SAC*, 2006.
- [10] K. Figly, J. Mendlingz et M. Strembecky, "Towards a Usability Assessment of Process Modeling Languages," 2009.

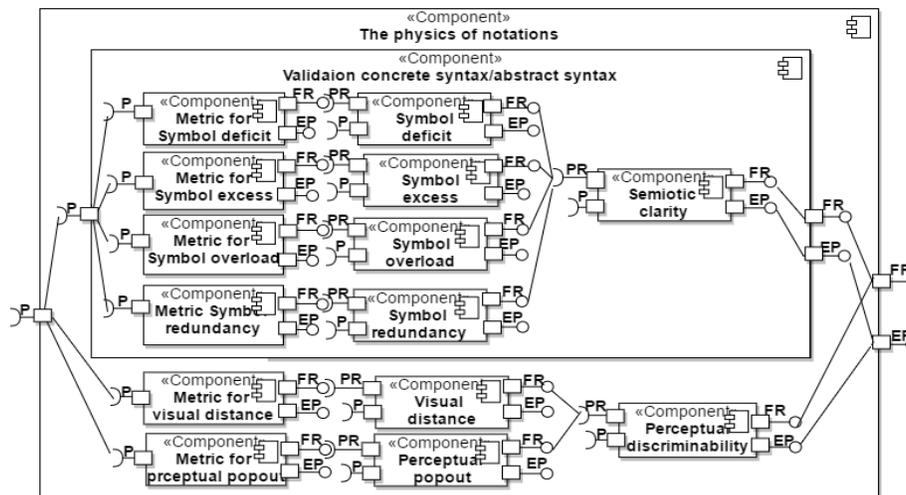


Figure 4. Evaluation process example

Keyword Search over Graph-structured Data for Finding Effective and Non-redundant Answers

Chang-Sup Park
 Department of Computer Science
 Dongduk Women's University
 Seoul, Korea
 cspark@dongduk.ac.kr

Abstract—In this paper, we propose a new method for keyword search over large graph-structured data to find a set of answers which are not only relevant to the query but also reduced and duplication-free. We define an effective answer structure and a relevance measure for the candidate answers to a keyword query on graph data. We suggest an efficient indexing scheme on relevant and useful paths from nodes to keywords in the graph. We present a top- k query processing algorithm to find relevant and non-redundant answers in an efficient way by exploiting pre-constructed indexes. We show by experiments using real datasets that the proposed approach can produce effective and non-redundant answers efficiently compared to the previous methods.

Keywords-graph data; keyword search; top- k query processing

I. INTRODUCTION

Recently, graph-structured data is widely used in various fields such as social networks, semantic web, linked open data, and knowledge bases. A relational database also can be considered a directed graph based on the foreign-key relationships among tuples. A graph data consists of nodes and edges, which can represent relationships among entities effectively. As the amount of data increases rapidly, an efficient and effective query system is much needed. Keyword search has been attracting a lot of attention since it allows users to express their information need using simple keywords [1-10].

Keyword search on graph data usually returns a set of connected sub-structures, showing that which nodes include query keywords and how they are inter-connected. Many approaches find minimal connected sub-trees as succinct answers to a given query [1-6, 8, 10]. Since there can be a significant number of answer sub-trees in a large graph data, a relevance scoring function is often used to rank candidate answers and select top- k ones having the highest relevance. There have been proposed several approaches based on the *distinct root semantics*, where the relevance of a sub-tree is computed as a function of the shortest paths from the root to the nodes containing query keywords. For each node in the graph, they choose at most one sub-tree rooted at the node as a candidate answer to the query [2, 3, 5, 10]. By reducing the number of candidates significantly, they can process top- k query over a large volume of data more efficiently than other approaches. It also facilitates exploitation of indexes on graph data to improve query performance [3].

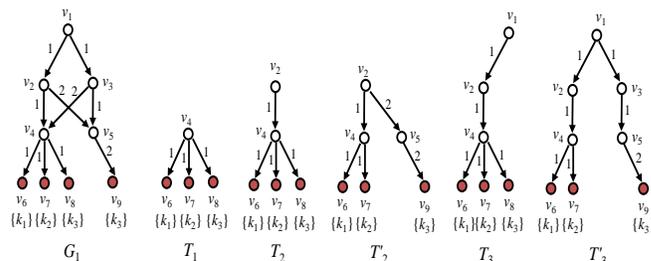


Figure 1. Reduced answers vs. non-reduced answers

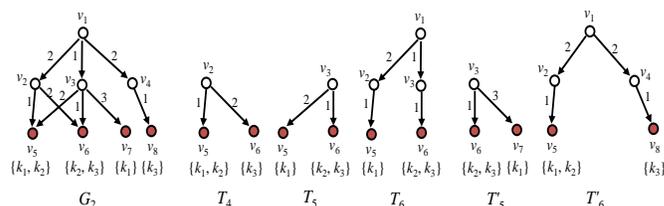


Figure 2. Duplicate answers vs. duplication-free answers

However, the previous methods have a common limitation; they can produce ineffective answers called a *non-reduced* tree and *duplicate* tree. The former is a sub-tree where the root node contains no query keyword and has only a single child node. For example, consider a directed weighted graph G_1 in Fig. 1 where nodes $v_6 \sim v_9$ contain keywords and edges are labeled with a weight representing distance between nodes. Given a keyword query $q = \{k_1, k_2, k_3\}$ over G_1 , five sub-trees shown in the figure can be answers to q since they have all the keywords in q in their nodes. Note that T_2 and T'_2 are rooted at the same node v_2 while having different nodes containing keyword k_3 , i.e. v_8 and v_9 , respectively. Since the distance from v_2 to v_8 is shorter than that from v_2 to v_9 , search methods usually select T_2 as the answer rooted at v_2 . However, it should be noted that T_2 is a non-reduced answer tree which has a smaller reduced answer T_1 as its sub-tree while T'_2 is a reduced answer. If T_1 is included in top- k results, selecting T'_2 instead of T_2 makes the search results more diverse even though the relevance score of T'_2 is lower than that of T_2 . It is also desirable to choose a reduced sub-tree T'_3 instead of a non-reduced tree T_3 as an answer rooted at node v_1 .

The other ineffectiveness in the previous approaches is that search results can include similar answer trees containing same set of content nodes for the query keywords. In Fig. 2, for instance, when a query $q = \{k_1, k_2, k_3\}$ is given on G_2 , the set of top-3 answers to the query based on the distinct root semantics is $\{T_4, T_5, T_6\}$. Note that these sub-trees have different root nodes but share the same set of content nodes $\{v_5, v_6\}$. If we select T'_5 and T'_6 instead of T_5 and T_6 , respectively, into top-3 answers, we can obtain results which are duplication-free and diverse in terms of the content nodes.

Top- k search results including many non-reduced and duplicate answer trees have drawbacks. First, similar and redundant answer trees decrease diversity of the search results and do not satisfy users who want to get various answers. Second, if an answer tree turns out to be irrelevant to the query, non-reduced or duplicate answer trees related with the answer would be also irrelevant to the query.

In this paper, we propose a new approach to keyword search over graph data which can produce not only relevant but also diverse results by searching top- k answers consisting of only reduced and duplicate-free answer trees. We suggest an extended indexing scheme on the selected paths in the graph and propose an efficient search algorithm exploiting the indices to find relevant and non-redundant answer trees.

II. RELATED WORK

Most previous approaches to keyword search on graph data find minimal sub-trees containing query keywords based on either Steiner-tree semantics [1, 4, 6] or distinct-root semantics [2, 3, 5, 10]. Under distinct-root semantics, sub-trees returned as query answers should be rooted at a distinct node. Thus, for each potential root node in the graph, only a single sub-tree having a minimal weight is considered a candidate answer, where the weight is defined by the sum of the lengths of the shortest paths from its root to keyword nodes. This semantics can deal with queries over a large graph data more efficiently than Steiner-tree semantics [11].

The Bi-directional Search proposed in [2] performs backward explorations of the graph starting from the nodes containing query keywords and also executes forward search from a potential root of an answer tree toward keyword nodes. However, it does not take advantage of any prior knowledge on the graph and depends on a heuristic activation strategy hence it shows poor performance on large graphs. BLINKS approach [3] proposes an efficient indexing scheme on the graph to speed bi-directional exploration with a good performance guarantee. It pre-computes the shortest paths and their distances from nodes to keywords in the graph and stores them in sorted inverted lists and a hash map. By exploiting indexes, it can avoid a lot of explorations in the graph and find top- k answers efficiently. For efficient search of a large graph data, [5] suggests creating and utilizing a multi-granular representation of graph data, and presents search algorithms on a multi-granular graph extended from BANKS [1] and Bi-directional Search. A recent study in [10] has proposed an extended answer structure with a new relevance measure and proposed an indexing and query processing scheme similar to BLINKS to produce effective and various top- k answers.

These approaches, however, have a common drawback of producing sub-trees that are non-reduced or duplicate in content nodes. Although graph exploration approaches such as BANKS and Bi-directional Search can detect and exclude such answer trees, an exponential number of sub-trees should be probed, resulting in severe performance overhead. BLINKS does not consider redundancies in answers, and even if a redundant sub-tree is detected, no other sub-tree rooted at the same node can be found since the method stores in its index only one optimal path from a node to keyword in the graph. That is, all alternative sub-trees sharing a root node are excluded from consideration. For example, BLINKS cannot produce answer trees T'_2 in Fig. 1 and T'_5 in Fig. 2 as alternatives to the redundant answers T_2 and T_5 . This can result in a set of limited and less relevant top- k answers to the query than the other approaches.

III. PROBLEM DEFINITION

A data graph $G(V, E)$ is a directed weighted graph where nodes in V contain keywords and edges in E have a weight representing distance between two incident nodes. The nodes containing a keyword k are called *keyword nodes* or *content nodes* regarding k and the set of those nodes is denoted by $V(k)$. The length of a directed path between two nodes in G is defined as the sum of the weights on edges in the path. Based on [11], we define an answer to a keyword query as follows.

Definition 1. Given a graph $G(V, E)$ and a query $q = \{k_1, k_2, \dots, k_l\}$ over G , an answer to q is a sub-tree T of G which contains a multiset $C = \{v_1, v_2, \dots, v_l\}$ of keyword nodes where $v_i \in V(k_i)$ ($1 \leq i \leq l$) and satisfies the following conditions: (a) T contains the shortest path from its root to each node in C , (b) all the leaf nodes of T belong to C , and (c) if the root of T has only one child, the root also belongs to C . \square

We denote an answer tree having a root node n and a multiset C of keyword nodes by $T(n, C)$. The shortest path from the root n to a keyword node v_i in C is called a *root-to-keyword path* for k_i and denoted by $n \rightarrow k_i$ or $n \rightarrow v_i$. The conditions in Definition 1 specify that answer trees should only have the nodes which are necessary and sufficient to connect their content nodes. In particular, condition (c) requires that answer trees should be *reduced*, i.e., the root of answer trees should have at least two child nodes or be a keyword node in itself. Assume that the root of an answer tree T has only one child and is not a keyword node. Then there exists a sub-tree T' in T which is reduced and has the same set of keyword nodes as T . Since T' is usually given the higher relevance score and preferred by a search method, T becomes a redundant answer to the query.

To find the most relevant answers to a given query, we propose a measure to the relevance of answer trees considering both content nodes and root-to-keyword paths in them. Given a node v having a keyword k , the relevance of v to k can be computed based on the TF-IDF weighting scheme which is popularly used in Information Retrieval [13]. For instance, adopting the weighting scheme used in Apache Lucene text search engine, relevance of v to k is defined by

$$rel(k, v) = \sqrt{tf(k, v)} \cdot \left(1 + \log\left(\frac{|V|}{|V(k)|+1}\right)\right)^2$$

where $|V|$ and $|V(k)|$ are the numbers of nodes in V and $V(k)$ and $tf(k, v)$ is the number of occurrences of k in v .

We also consider the length of the path from the root to each keyword node to measure structural relevance of answer trees. We consider that given an answer tree T , the shorter the distance $dist(n, v_i)$ from its root n to a keyword node v_i , the more relevant to the query the tree T is. Now, the relevance scoring function for answer trees is defined as follows.

Definition 2. Given an answer tree $T(n, \{v_1, v_2, \dots, v_l\})$ for a query $q = \{k_1, k_2, \dots, k_l\}$, the relevance of T to the query q is

$$rel(T, q) = \sum_{1 \leq i \leq l} rel(n, k_i, v_i)$$

where

$$rel(n, k_i, v_i) = \frac{rel(k_i, v_i)}{r_{max}} \cdot \left(1 + \log\left(\frac{1}{dist(n, v_i) + 1}\right)\right)$$

and r_{max} is the maximal value of $rel(k, v)$ for all keyword terms k and nodes v in G . \square

Note that, in the above definition, relevance $rel(T, q)$ is computed as the sum of relevance $rel(n, k_i, v_i)$ defined on each keyword node and root-to-keyword path in T .

In this paper, we search for the answer trees which are not only reduced but also duplication-free in regard of their content nodes. It means that the answer trees should have different sets of content nodes, as well as should be rooted at distinct nodes. Based on this semantics, we aim to finding k most relevant answers to the query using the relevance function defined above.

IV. PROPOSED METHOD

A. Indexing Scheme

To enable efficient search of top- k answers in a large graph data, we propose an indexing scheme for selected node-to-keyword paths in the graph, based on that of BLINKS [3]. It pre-computes the most relevant and useful paths using the relevance function proposed in Definition 2 and stores them in the index consisting of the following components.

- *KNList(Keyword-Node Lists)* is a set of inverted lists $KNList(k)$ defined for each keyword k in the graph. It stores the most relevant node-to-keyword path from each node to a keyword node for k . Specifically, let $P(n, k) = \{n \rightarrow v_i \mid v_i \in V(k)\}$ for a node n and keyword k , and $p_m(n, k)$ be the optimal path in $P(n, k)$ which has the highest value of $rel(n, k, v_i)$. $KNList(k)$ stores entries representing $p_m(n, k)$ for all nodes n in the graph. Entries are quadruples (n, v_m, f_m, r_m) , where v_m denotes the keyword node containing k , f_m is the *first node* except the start node n (i.e., the next node of n), and r_m is the relevance of the path, $rel(n, k, v_m)$. The entries in the list are sorted in a decreasing order of relevance, which enables efficient search of most relevant paths for keyword k .
- *NKMap(Node-Keyword Map)* is a hash table to store information about the most relevant paths for each pair (n, k) of node and keyword in the graph. It stores entries of a

pre-defined number of n -to- k paths with the highest relevance, including the optimal path $p_m(n, k)$. The entries are triples (v_i, f_i, r_i) where the fields denote the same as those in $KNList(k)$.

- *NKMap_s* is a secondary hash table to store information about an alternative node-to-keyword path for all pairs of node n and keyword k in the graph. It is the most relevant path which has the first node different from that of the optimal path $p_m(n, k)$. This index is used to find the most relevant reduced answer trees.

B. Query Processing Algorithm

Our query processing model is based on the Threshold Algorithm [12] which is used to evaluate top- k queries on multi-dimensional data such as multimedia objects. Algorithm 1 shows the sketch of our search algorithm, which uses a priority queue Q_t called top- k queue to maintain top- k candidate answers.

Algorithm 1. Keyword Search

Input: a keyword query $q = \{k_1, k_2, \dots, k_l\}$, $k \in \mathbb{Z}^+$

Output: a set of top- k answer trees for q

- 1: a priority queue Q_t and a set C of *nodeID*'s by ϕ
 - 2: $curRel[i] \leftarrow 0.0$ and $V[i] \leftarrow null$ for all $i \in [1, l]$
 - 3: Let $L(q) = \{KNList(k_i) \mid k_i \in q \ (1 \leq i \leq l)\}$.
 - 4: **while** an entry exists in a list in $L(q)$ **do**
 - 5: Select a list L_i in $L(q)$ in a round-robin manner.
 - 6: Read an entry (n, v, f, r) at the current position in L_i .
 - 7: $curRel[i] \leftarrow r$
 - 8: **if** $n \notin C$ **then**
 - 9: $V[i] \leftarrow (v, f, r)$
 - 10: **for-each** $k_j \in q$ such that $j \neq i$ **do**
 - 11: Look up the first entry (v_j, f_j, r_j) with key (n, k_j) in $NKMap$.
 - 12: **if** the entry was found **then** $V[j] \leftarrow (v_j, f_j, r_j)$
 - 13: **else** goto line #21
 - 14: **if** $T(n, V)$ is a non-reduced answer **then**
 - 15: $V \leftarrow findReducedAnswer(n, V, q)$
 - 16: **if** $V \neq \phi$ and $T(n, V)$ is a duplicate answer **then**
 - 17: $V \leftarrow findUniqueAnswer(n, V, q)$
 - 18: **if** $V \neq \phi$ **then** $Q_t \leftarrow Q_t \cup \{(n, V)\}$
 - 19: $C \leftarrow C \cup \{n\}$
 - 20: **if** $|Q_t| = k$ and $rel_k \geq \sum_{1 \leq i \leq l} curRel[i]$ **then break**
 - 21: Derive top- k answer trees from the top- k entries in Q_t .
-

Given a query $q = \{k_1, k_2, \dots, k_l\}$, let $L(q)$ be the set of keyword-node lists $KNList(k_i)$ for all keywords k_i in q . The algorithm performs sequential scan on the lists in $L(q)$ in parallel (line 5~6). Whenever a new entry is read from a list, its relevance value is stored in an array $curRel$ (line 7). If an entry (n, v, f, r) regarding an optimal path from a node n is first retrieved from $L(q)$, entries of the optimal paths $p_m(n, k_j)$ for all the other keywords k_j in q are looked up in $NKMap$ and aggregated into an array V (line 8~13). If all the optimal paths are found, an optimal answer tree rooted at n can be derived. We examine whether it has a reduced form and has a unique set of content nodes compared to the other candidates in top- k queue. If it does not, we seek an alternative reduced and unique answer tree using algorithms which will be detailed later (line

14~17). The result tree is stored in Q_i if it is one of the k most relevant found yet (line 18). Since the entries in each list are sorted in a decreasing order of relevance, the sum of the values in $curRel$ can serve as an upper bound of relevance of the answer trees which have not been found yet. Thus, the algorithm can terminate safely with the correct top- k answers in Q_i if the condition in line 20 is met, where rel_k is the relevance of the k -th answer tree in Q_i .

C. Finding Reduced and Unique Answer Trees

Given a potential root node n , Algorithm 1 searches for an optimal answer tree consisting of the best root-to-keyword paths for each query keyword. If the optimal answer is a non-reduced tree where its root has only one child but is not a keyword node, the first nodes of l root-to-keyword paths are the same to the child of the root. Thus, we can find if the optimal tree is reduced or not by examining the first nodes of all the root-to-keyword paths retrieved from $NKMap$. However, it should be considered that if the root contains all the query keywords and is selected as keyword nodes for them, the root itself can be a reduced answer tree.

Assuming that $T(n)$ is a non-reduced answer tree rooted at node n , if there are multiple reduced answer trees rooted at the same node n , we should select one with the highest relevance score as an alternative to $T(n)$. For keyword k_i in q , let $p_a(n, k_i)$ be the path from n to a node v in $V(k_i)$ which has the first node different from that of $p_m(n, k_i)$ and has the highest score of $rel(n, k_i, v)$. Also suppose that $T_i(n)$ be the sub-tree obtained by replacing the optimal path $p_m(n, k_i)$ with $p_a(n, k_i)$ in $T(n)$. Note that $T_i(n)$ is a reduced answer to q since the first node of $p_a(n, k_i)$ is not equal to those of the other root-to-keyword paths $p_m(n, k_j)$ for keywords k_j in q ($j \neq i$). Now, among l alternative sub-trees $T_i(n)$, the one with the highest relevance is the best reduced answer tree rooted at n .

Algorithm 2 exploits $NKMap_s$ index proposed in Section 4.1, which pre-computes and stores the optimal alternative paths $p_a(n, k)$ for all pairs of node n and keyword k in the graph. Given a non-reduced answer $T(n)$, it first looks up information on alternative paths from n to all the query keywords in $NKMap_s$ (line 2~4). An optimal reduced answer tree can be easily obtained by selecting such a keyword k_i that difference between $p_m(n, k_i)$ and $p_a(n, k_i)$ is the smallest and replacing $p_m(n, k_i)$ with $p_a(n, k_i)$ in $T(n)$ (line 7~8).

Algorithm 2. *findReducedAnswer*

Input: a nodeID n , an array V of $(nodeID, nodeID, rel)$'s, a query q

Output: an array $V[1..l]$ of $(nodeID, nodeID, rel)$'s

- 1: Let $A[1..l]$ be an array and $A[i] \leftarrow null$ for all $i \in [1, l]$.
 - 2: **for-each** $k_i \in q$ **do**
 - 3: Look up entry (v_i, f_i, r_i) with key (n, k_i) in $NKMap_s$.
 - 4: **if** the entry was found **then** $A[i] \leftarrow (v_i, f_i, r_i)$
 - 5: **if** $A[i] = null$ for all $i \in [1, l]$ **then return** ϕ
 - 6: **else**
 - 7: Find $i \in [1, l]$ such that $(V[i].rel - A[i].rel)$ is minimal.
 - 8: $V[i] \leftarrow A[i]$
 - 9: **return** V
-

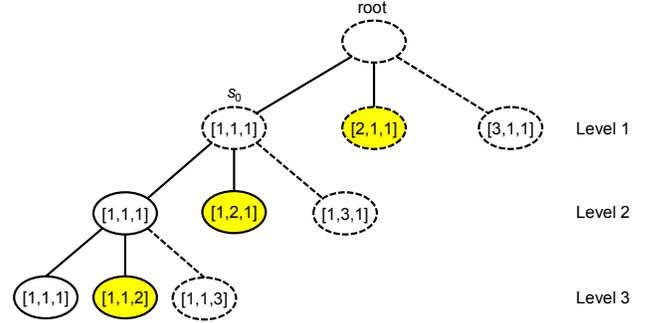


Figure 3. State space search performed by Algorithm 3

Now, we consider finding top- k answer trees which are duplication-free in regard of content nodes. An answer tree $T(n, C)$ is duplicate if and only if (a) its relevance score is greater than that of the k -th candidate answer tree in Q_i and (b) there exists an answer tree $T(m, C)$ in Q_i which contains the same set of content nodes as $T(n, C)$ and has no smaller relevance score than $T(n, C)$. Given a duplicate tree T , we should find another answer tree $T(n, C')$ which is rooted at the same node n as T and has a set of content node different from all the other answers in Q_i . Assuming that the graph has p root-to-keyword paths from n to each keyword in the query of size l , p^l answer trees rooted at n can be derived from the combinations of the paths. To find the optimal one which is not duplicate and has the highest relevance score efficiently without generating all the possible answers, we suggest a state space search algorithm based on branch-and-bound strategy, shown in Algorithm 3. It exploits $NKMap$ index which has information on p most relevant paths from a node to keyword in a decreasing order of relevance. As shown in an example in Fig. 3, the search space forms a tree of states each of which represents a combination of root-to-keyword paths for query keywords and derives an answer tree. A state at level i selects one of p paths for k_i ($1 \leq i \leq l$) and has the optimal paths for all keywords k_j where $i < j \leq l$. For the other keywords, it inherits the selection of paths from its parent state. In Fig. 3, the indexes of the selected paths are presented in the states. We can see that an answer tree represented by a state s has no smaller relevance score than those derived from the descendent states of s in the states tree. Also note that sibling states choose a different path to the same keyword in a decreasing order of relevance from left to right hence the answer tree from s has no smaller relevance than those derived from the right sibling states of s . Considering these features, Algorithm 3 explores the search space in an efficient way by pruning a large portion of unnecessary states.

In Algorithm 3, a priority queue Q_s is used to store states to be explored, starting from the initial state which has the optimal paths for all the query keywords (line 2). At each stage, a state e with the highest relevance score is selected from the queue (line 4~5). If the score of e is greater than that of the best state found yet, denoted by LB , the next sibling state s of e is generated and investigated (line 8~16). If the score of s is no greater than LB , all of its descendent states can be safely excluded from further exploration. If s derives a reduced and unique candidate answer tree and its score is greater than LB , it is considered a new best solution state (line 12~14). Otherwise, it is stored in Q_s . If e is not a leaf state, the first child of e is

generated and the above process is repeated on it (line 17~19). Fig. 3 shows the states generated in the first round of the outermost loop in Algorithm 3. During the best-first search, if a state selected from Q_s derives no better answer trees than the best solution state found yet, the search can terminate and return the best answer tree (line 6).

Algorithm 3. *findUniqueAnswer*

Input: a *nodeID* n , an array V of (*nodeID*, *nodeID*, *rel*)'s, a query q

Output: an array $V[1..l]$ of (*nodeID*, *nodeID*, *rel*)'s

```

1:  $UB \leftarrow score(T(n, V))$ ,  $LB \leftarrow rel_s$ ,  $bestSolution \leftarrow \phi$ 
2: a priority queue  $Q_s \leftarrow \{s_0\}$ , where  $s_0$  is an initial state.
3: while there exists a state in  $Q_s$  do
4:    $e \leftarrow$  a state in  $Q_s$  whose score is maximal
5:    $Q_s \leftarrow Q_s / \{e\}$ 
6:   if  $score(e) \leq LB$  then break
7:   loop
8:     if  $e$  has the next sibling state then
9:       Generate the next sibling state  $s$  of  $e$ .
10:      if  $score(s) > LB$  then
11:        if  $score(s) \leq UB$  then
12:          if  $s$  is a solution state then
13:             $bestSolution \leftarrow s$ 
14:             $LB \leftarrow score(s)$ 
15:          else  $Q_s \leftarrow Q_s \cup \{s\}$ 
16:          else  $Q_s \leftarrow Q_s \cup \{s\}$ 
17:        if  $e$  is a non-leaf state then
18:          Generate the first child state  $c$  of  $e$ .
19:           $e \leftarrow c$ 
20:        else break
21: if  $bestSolution \neq \phi$  then
22:   for-each  $k_i \in q$  do
23:      $V[i] \leftarrow$  an entry  $(v_i, f_i, r_i)$  which is looked up in  $NKMap$ 
with the key  $(n, k_i)$  and selected by  $bestSolution$ 
24:   return  $V'$ 
25: else return  $\phi$ 

```

V. PERFORMANCE EVALUATION

We evaluate effectiveness and efficiency of the proposed method by experiments using real graph data. We compare the performance of our method with BLINKS [3], which uses path indexes similar to our method, and a modified version of it, called *BLINKS-N*, which detects and excludes non-reduced or duplicate sub-trees from the candidates. We implemented two versions of the proposed method; *Reduced* method only finds top- k reduced answers while *Reduced&Unique* method produces both reduced and duplication-free answers. All the algorithms are implemented in Java. We used JGraphT¹ library to construct in-memory graph structures and compute the shortest paths between pairs of nodes. We exploited Apache Lucene² library to extract keywords from nodes in the graph and compute the relevance of the nodes to keyword terms.

As for the test graph datasets, we use a geographic data Mondial³ and a movie database IMDB⁴. From Mondial, we selected a subset of entities and relationships to build a graph including 6,431 nodes, 19,951 edges, and 15,815 keyword

TABLE 1. TEST QUERIES

Mondial		IMDB	
Query	Keyword list	Query	Keyword list
Q1	caldera, lake, america	Q11	drama, sports, competition
Q2	cape, gulf, africa	Q12	friendship, love, marriage
Q3	vienna, donau, alps	Q13	emperor, war, battle
Q4	lake, quebec, canada	Q14	hitchcock, mystery, thriller
Q5	himalaya, india, pakistan	Q15	police, crime, violence
Q6	river, minnesota,	Q16	human, vampire, fight
Q7	city, desert, california	Q17	thriller, murder, crime
Q8	lake, michigan, ontario	Q18	natural, disaster, war
Q9	island, vancouver, seattle	Q19	president, politics, drama
Q10	alaska, arctic, sea	Q20	accident, explosion, crash

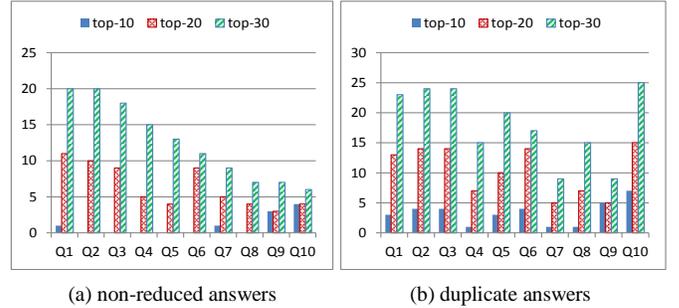


Figure 4. Number of non-reduced and duplicate answers by BLINKS

terms. In IMDB database, we derived a large graph consisting of 831K nodes, 2.82M edges, and 303K keyword terms. For the sake of simplicity and efficiency of experiments, we assume that all edges have the same weight of 1 and only use the node-to-keyword paths the length of which are no longer than 5. Our experiments have been conducted on a LINUX server having 10 1.7GHz hexa-core CPUs and 32GB RAM.

Table 1 shows a subset of the test queries used in experiments. We executed the search methods to find top- k answers to the queries. Fig. 4 shows top-10, top-20, and top-30 answers to the queries on Mondial obtained by BLINKS. It shows that the results include respectively 1, 6, and 13 non-reduced answers and 3, 10, and 18 duplicate answers on average while the proposed method has returned no non-reduced or duplicate answer at all.

Fig. 5 presents relevance scores of top-30 answers to the queries on Mondial obtained by each method. It shows that the results by our method have lower relevance scores than those by BLINKS, but *Reduced* method achieves higher relevance than *BLINKS-N* for all test queries. Fig. 6 shows that the average relevance score of the results by *Reduced* is about 6.6% lower than the results by BLINKS, but about 3.1% higher than the answers obtained by *BLINKS-N*. This indicates that even though our method replaces non-reduced answers with reduced ones which may have lower relevance scores, it can produce more effective results than *BLINKS-N*.

Fig. 6 also shows that average execution times of two versions of our method increase by about 37.8% and 52.7% respectively compared to BLINKS, due to the overhead

¹ <http://www.jgrapht.org/>

² <http://lucene.apache.org/java/docs/index.html>

³ <http://www.dbis.informatik.uni-goettingen.de/Mondial/>

⁴ <http://www.imdb.com/>

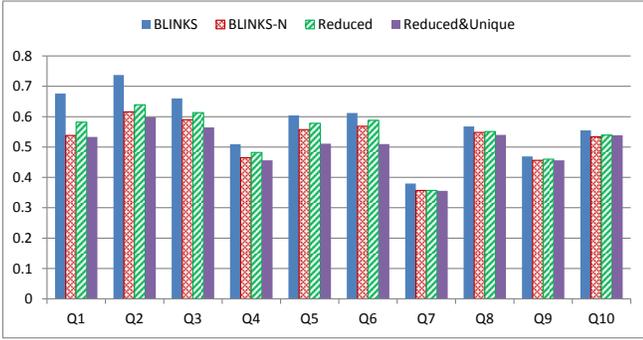


Figure 5. Relevance of top-30 answers

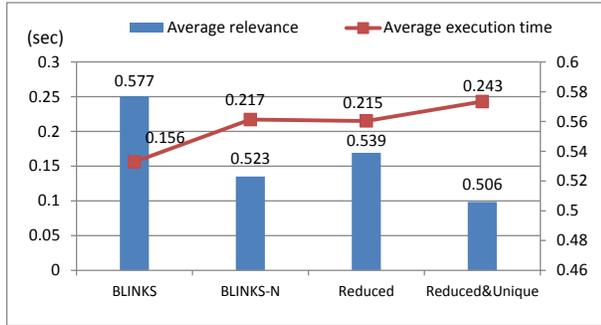


Figure 6. Average relevance and execution time

occurred by additional search for the optimal reduced and duplication-free answers. However, we can observe that *BLINKS-N* is also degraded by 39.1% and our *Reduced* method slightly outperforms it.

In Fig. 7, we compare execution performance of the best-first state search strategy employed in *Reduced&Unique* method with that of a naïve approach to conducting brute-force search to find optimal duplication-free answers, based on the search result for the test queries on IMDB dataset. Fig. 7-(a) shows the number of states, i.e., answer trees explored in each method, and Fig. 7-(b) presents their execution time. We observe that our method achieves performance improvement about 62.0% in the number of states generated and 65.9% in execution time.

VI. CONCLUSION

In this paper, we proposed a new approach to keyword search over graph data to find answers which are not only relevant to the query but also reduced and duplication-free. We suggested an efficient indexing scheme to index relevant paths between nodes and keywords in the graph, and proposed a top-*k* query processing algorithm to find the most relevant non-redundant answers in an efficient way by exploiting the pre-constructed indexes. By producing non-redundant and relevant answer trees, it can provide users with diverse and effective query results and thus satisfy the users' information need. We showed by experiments using a real graph dataset that our approach is effective and efficient for a large graph data.

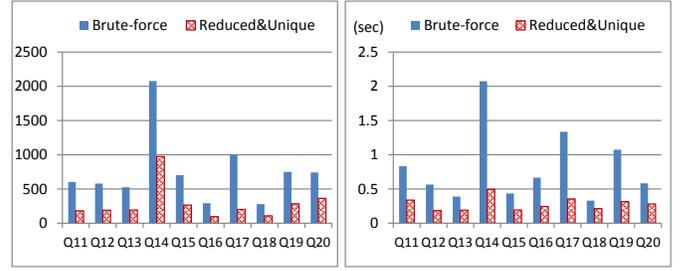


Figure 7. Performance of the state search algorithm to find duplication-free answers

ACKNOWLEDGMENT

This work was conducted in part while the author was visiting the Department of Computer Science and Engineering, University of California, San Diego, USA. The author would like to thank Prof. Yuanyuan Zhou for inviting him to the UC San Diego and for providing research environment.

REFERENCES

- [1] G. Bhalotia, A. Hulgeri, C. Nakhe, S. Chakrabarti, and S. Sudarshan, "Keyword searching and browsing in databases using BANKS," In Proc. of IEEE 18th Int. Conf. on Data Engineering, pp.431-440, 2002.
- [2] V. Kacholia, S. Pandit, S. Chakrabarti, S. Sudarshan, R. Desai, and H. Karambelkar, "Bidirectional expansion for keyword search on graph databases," In Proc. of the 31st Int. Conf. on Very Large Data Bases, pp.505-516, 2005.
- [3] H. He, H. Wang, J. Yang, and P. S. Yu, "BLINKS: ranked keyword searches on graphs," In Proc. of 2007 ACM SIGMOD Int. Conf. on Management of Data, pp.305-316, 2007.
- [4] B. Ding, J. X. Yu, S. Wang, L. Qin, X. Zhang, and X. Lin, "Finding top-k min-cost connected trees in databases," In Proc. of IEEE 23rd Int. Conf. on Data Engineering, pp.836-845, 2007.
- [5] B. B. Dalvi, M. Kshirsagar, and S. Sudarshan, "Keyword search on external memory data graphs," Proceedings of the VLDB Endowment, Vol.1, No.1, pp.1189-1204, 2008.
- [6] K. Golenberg, B. Kimelfeld, and Y. Sagiv, "Keyword proximity search in complex data graphs," In Proc. of 2008 ACM SIGMOD Int. Conf. on Management of Data, pp.927-940, 2008.
- [7] L. Qin, J. X. Yu, L. Chang, and Y. Tao, "Querying communities in relational databases," In Proc. of IEEE 25th Int. Conf. on Data Engineering, pp.724-735, 2009.
- [8] T. Tran, S. Rudolph, P. Cimiano, and H. Wang, "Top-k exploration of query candidates for efficient keyword search on graph-shaped data," In Proc. of IEEE 25th Int. Conf. on Data Engineering, pp.405-416, 2009.
- [9] M. Kargar and A. An, "Keyword search in graphs: finding r- cliques," Proceedings of VLDB Endowment, Vol.4, No.10, pp.681-692, 2011.
- [10] C. Park and S. Lim, "Efficient processing of keyword queries over graph databases for finding effective answers," Information Proc. and Mgmt, Vol.51, No.1, pp.42-57, 2015.
- [11] J. X. Yu, L. Qin, and L. Chang, "Keyword search in relational databases: a survey," Bulletin of the IEEE CS Technical Committee on Data Engineering, Vol.33, No.1, pp.67-78, 2010.
- [12] R. Fagin, A. Lotem, and M. Naor, "Optimal aggregation algorithms for middleware," Journal of Computer and System Sciences, Vol.66, No.4, pp.614-656, 2003.
- [13] S. Butcher, C. Clarke, and G. Cormack, Information retrieval: implementing and evaluating search engine. MIT Press, 2010.

Informed and Timely Business Decisions – A Data-driven Approach

Veera Tadikonda and Daniela Rosca

Computer Science and Software Engineering Department Monmouth University
West Long Branch, NJ, USA

{veera.tadikonda@gmail.com, drosca@monmouth.edu}

Abstract— One of the main characteristics of business rules is their propensity for frequent change, due to internal or external factors to an enterprise. As these rules change, their immediate dissemination across people and systems in an enterprise becomes vital. The delay in dissemination can adversely impact the reputation of the enterprise, and cause significant loss of revenue. The current BRMS are often maintained by the IT group within a company, therefore the modifications of the BRs intended by executive management would not be instantaneous, since they have to be coded, and tested before being deployed. Moreover, the executives might not have the possibility to take the best decisions, without having the benefit of analyzing historical data, and quickly simulating what-if scenarios to visualize the effects of a set of rules on the business. Some of the systems that provide this functionality are prohibitively expensive. This paper addresses these challenges by using the power of Big Data analysis to source, clean and analyze historical data that is used for mining business rules, which can be visualized, tested on what-if scenarios, and immediately deployed without the intervention of the IT group. The proposed approach is instantiated in this paper by using open source components to mine stop loss rules for financial systems.

Keywords: *Business Rules, Decision Support, Big Data, Rule Mining*

I. INTRODUCTION

Using a single piece of information to derive decisions could lead to polarized results and unprofitable decisions. To overcome this limitation, information from different sources, quantitative as well as qualitative has to be utilized. With today's overwhelming number of information sources, one cannot neglect the benefits they can bring to a business by integrating the information and extracting a meaning from it. Based on the recent advances in Big Data analytics, businesses can use historical and market data to gain competitive advantages. These methods prove to be beneficial to software engineering tasks as well, such as identifying and classifying requirements and mining sentiment analysis from App reviews [1], [2], automatic requirements elicitation from feedback comments [3], mining user comments for helping requirements evolution[4], or extracting features from product descriptions to recommend features implementation for software product lines [5].

Another major area of applying mining techniques in software engineering has been process mining [6] for tasks such as discovering processes from event logs, testing the

conformance of processes, improvement of business processes, or mining user's intentions in intentional process models [7], [8]. One very important component of business processes has been the treatment of business rules (BRs). In today's highly agile world, where businesses need to quickly adapt to market changes, business processes have to dynamically react, and avoid locking processes in hard to change IT solutions. vonHalle [9] and Ross [10] have long been advocating the business rules approach, that claims that all BRs in an enterprise should be collected in a centralized business rules management system (BRMS). However, these BRs need to be created and verified by the business experts, who might not be familiar with formal languages, and hence their dependence on the IT staff. In order to alleviate this problem, a couple of solutions have been proposed, such as the OMG standard, Semantics of Business Vocabulary and Rules – Structured English (SBVR-SE) [11], a controlled natural language for business rules specifications (RuleCNL) [12], or the transformations of the SBVR specifications into UML/OCL [13], or BPMN metamodel [14]. However, Lucie performed a state of the art analysis of business rules languages in [15], and concluded that most of them are difficult to use by business experts.

To avoid the need of using formal or semi-formal languages for expressing business rules, we propose an approach where the business rules are mined from existing historical data and market data, and are presented to the business user in an intuitive format for verification and immediate deployment. This solution is close to [16], which advocates the induction of BRs from statistical data using decision trees learning. With the recent advances in Big Data and Machine Learning, the abilities to create rules have been considerably expanded. The approach proposed in this paper covers not only the creation and modification of BRs, but also supports the dynamic decision making needs of a business user, by presenting in a visual, easy to grasp way, the historical and current market data, and allowing the simulation of what-if scenarios to understand the impact of a set of rules on the business. Whenever the business user approves new rules or changes existing ones, they can be immediately deployed using an operational rule engine, as opposed to approaches mentioned above, which stop at the specification of business rules in a modeling language, without the benefit

of observing their effects on the business. This way, an agile reaction of the business to the market is made possible.

The remainder of the paper is structured as follows: Section II describes the process and architecture that support the proposed approach. Section III presents the method for mining business parameters out of historical data, while Section IV describes how these parameters are used for mining BRs conditions. Section V shows the steps for generating the rules, while Section VI presents the deployment, execution, and feedback of the rules' execution. Finally, the paper concludes with a summary, limitations and plans for extending this work.

II. PROCESS AND SOLUTION ARCHITECTURE

A. Process

The solution proposed in this paper allows the business user to interactively and dynamically create or change business rules, using historical and market data. The process (as shown in Fig.1) is started with the classification phase, where the variables used in the subsequent rules are identified, based on either a supervised or unsupervised learning algorithm, depending on whether the variables are well known for the business user, or they need to be discovered from a data set, or a new source of data needs to be mined. Using the variables identified in the previous step, and the trends from the available data set, the system identifies conditions that can be applied to the active data to create corresponding rules. The business user has the option to vary the range of the variables that are under control by the business, dynamically visualizing the new conditions. Once the conditions are decided upon, the system creates corresponding rules. The user has the ability to choose all or some of the generated rules for deployment. Alternatively, the user can create his own rules with autosuggestions from the system. The created rules are applied to the active data, to generate actionable decision rules, specific to the domain of application. Before deciding on the best rules, the business user can visualize the effects of applying a set of rules in what-if scenarios. Once the best rule set is identified, it is deployed for execution on the active data set, using a rule engine. The results of the execution are fed back into the system's persistent storage, to be used in later decision processes.

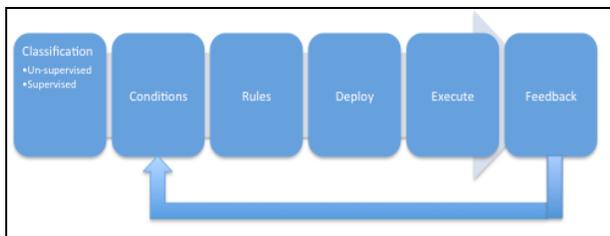


Figure 1. Process for dynamic BRs creation, modification and deployment

The domain of application for this paper is the stop-loss rules used in financial systems. Stop-loss is a technique of safeguarding the losses by the investor. The investor chooses to sell the stock if it falls to a certain price, such that the amount lost is limited to his risk capacity. The scenario that will be implemented in this paper refers to making recommendations for selling or buying a certain stock. It shadows the following

procedure: using a standard stock movement prediction model, build an exposure chart, build a historical chart, create stop-loss rules and generate upside threshold messages for selling or buying. The fundamentals and market information from various sources are collated to predict an outcome for the stocks. These outputs/ predictions/ recommendations are converted to actionable rules. This scenario follows the steps of algorithmic trading.[17]

B. System Architecture

The approach proposed here relies on a generic architecture that is composed of four building blocks:

1. *Data collation and analysis* – to capture heterogeneous data from multiple sources and analyze it.
2. *Data transfer* – a typical Extract Transfer Load (ETL) component that extracts data from multiple sources and transforms it to be stored in an appropriate format in a NoSQL data store.
3. *Aggregation/Computation* – processes data from the NoSQL data store using Big Data tools, in order to extract business parameters and rule conditions.
4. *Rule management and deployment* – to support the creation, modification and deployment of the rules generated to a rules engine, in order to trigger the execution of relevant actions. It also helps to capture the feedback of the rules execution such that it can be used in future rule generation iterations.

An instantiation of this generic architecture is described next for mining stop loss rules for financial systems.

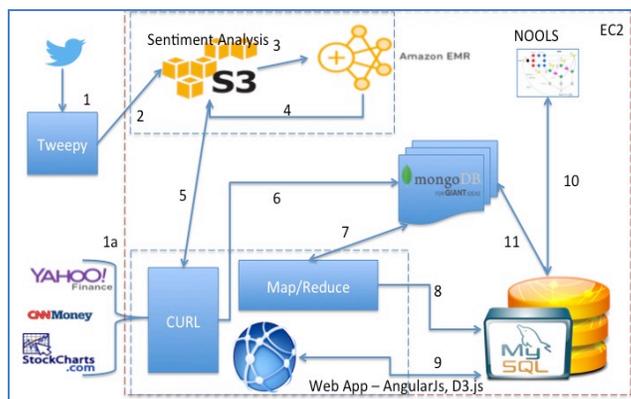


Figure 2. Instantiated Architecture Diagram

The architecture shown in Fig. 2 is using data from multiple sources, such as current portfolio, historical trades, Tweets (for determining sentiment and market information), Stock, Sector, and Index quotes from Yahoo, CNNMoney, and Stockcharts, respectively. Tweepy is used for extracting tweets pertaining to the stocks in the portfolio and provide them to Amazon EMR (Elastic Map-Reduce) to analyze the number of positive, negative, and neutral tweets. The results of the analysis are stored into Amazon S3 (simple storage service). Using CURL, the sentiment analysis information, as well as the stock prices and trade information, are stored into MongoDB. The information from this document storage is used in a Map-Reduce script to generate scores for stock, sector, and index quotes, as well as sentiment scores, necessary for mining conditions. The output from Map-

Reduce is stored on a MySQL transactional database, as conditions, together with rules, historical trades and portfolio data. The trade rules are being deployed in real-time using Nools (the Rete algorithm based rule engine) and the current data set. The output from the execution engine is circled back to the system's data stores to improve the accuracy of its suggestion algorithm in future iterations. The interfaces between the architectural components are shown in Table 1.

Id	Source	Purpose
1	Twitter Feed to Tweepy	Fetch tweets from Twitter
1a	Stock, Sector and Index quotes from Yahoo, CNNMoney, Stockcharts	Fetch Stock, Sector and Index quotes
2	Tweets	Store tweets into Amazon S3 storage
3	Tweets from Amazon S3 to Amazon EMR	Process Tweets using Amazon EMR
4	Amazon EMR output	Sentiment analysis output from Amazon EMR
5	Amazon S3 Sentiment analysis output	Using CURL to fetch Sentiment analysis output from Amazon S3
6	CURL to store Stock and Sentiment	Store Stock and Sentiment data into MongoDB
7	Data for Map-Reduce function	Interchange data between MongoDB and Map-Reduce function to generate scores
8	Map-Reduce output	Push Map-Reduce output into MySQL
9	Web Application to MySQL	For data interchange between Web application and MySQL
10	MySQL to NOOLS	Send and receive Rules and Execution information between MySQL and NOOLS
11	MySQL to MongoDB	Send execution results from MySQL to MongoDB

Table 1. Interfaces between the architectural components.

The entire architecture runs on the Amazon EC2 (Elastic Computer Cloud) platform, to allow the system to be used simultaneously by multiple users on different instances. The solution is built by integrating various open source applications in a web application. The design approach for this solution has kept customization to a minimal level, such that it can be applied across industries. Also, the usage of loosely coupled components in the architecture makes the solution implementable across different platforms and scenarios, and makes it possible that a part of the solution is in demilitarized zone, and the rest of the application, behind firewall. Next, we describe how the components of this architecture contribute to each phase of the process model shown in Figure 1.

III. CLASSIFICATION - MINING BUSINESS PARAMETERS

In this phase, related business parameters (variables) are determined from the historical trades data, current portfolio, and market data (see Figure 3), using a supervised or unsupervised machine-learning algorithm. For this application, since the variables are already known (stock price, sector score, index score, overall score and sentiment score), a linear regression supervised learning algorithm has been utilized. The variables are chosen to get a broad perspective of the stock price changes, since a stock price is generally dependent on the following factors:

- Company news and performance (reflected by the Twitter feed – Sentiment score)

- Industry performance (reflected by Sector score)
- Investor sentiment (Bull/bear market – reflected by Index score)

Based on these scores, the Overall Stock Score can be calculated as will be shown in Section V.

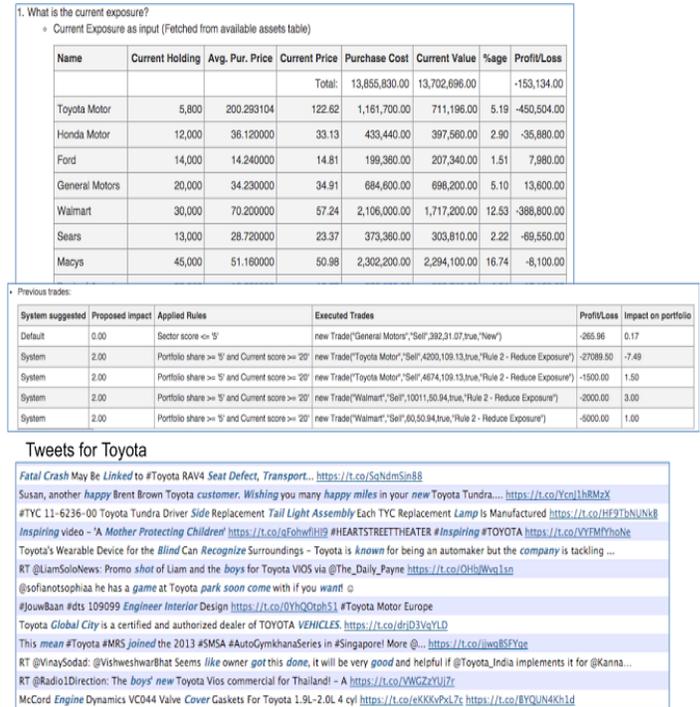


Figure 3. A snapshot of current portfolio data, historical trades, and market data

The linear regression algorithm is used to get the best-fit line for each of the above scores. To get the best-fit line, the linear regression formula of $y = a + bx$ is used, where “a” is the intercept and “b” is the slope. The r^2 provides the correlation coefficient, with values between -1 to +1. The inclination towards +1 denotes a strong correlation between the variables being tested, and thus for this application, a coefficient value above +0.5 is taken as a strong indicator of the relevance of the variables on the stock prices at that point in time. From the variables mentioned above, only the Sector score and Overall score show a strong correlation with the stock price, with r^2 values of 0.98 (as seen in Figure 4).

IV. DETERMINING RULES CONDITIONS

The relevant variables are used as inputs to another regression algorithm to find the best-fit line that will help identify the optimal values for them, values that will be used in determining the rules' conditions.

The pentagon chart (radar chart shown in Figure 4) plots the current scores of five axes that influence the risk determination for a stock. The five axes comprise of Alpha, Beta, Sharpe ratio, Risk tolerance, and Time horizon.

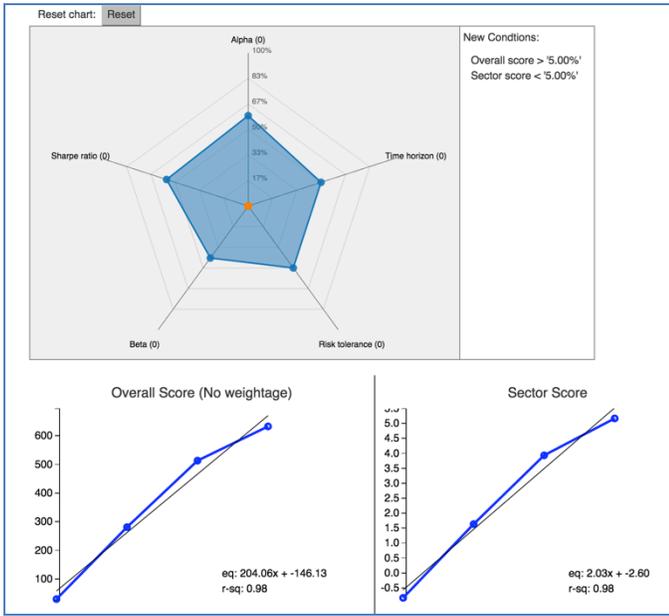


Figure 4. Radar chart visualization of new rules conditions

The “New Conditions” window shows the details of the newly mined conditions (for the sample set that we used, the mined conditions were “Overall Score > 5%” and “Sector Score < 5%”). These conditions are the ones that are carried to the next step, the rules generation. The conditions resulting here are generated by the system based on the linear regression algorithm, and further modified by the user by dynamically adjusting the arm values on the radar chart.

Of the five axes of the radar chart, Alpha, Beta and Sharpe ratio are driven by the market conditions. The investor/portfolio manager controls only the Risk Tolerance and Time Horizon. A portfolio manager can reduce or increase Risk. In addition to the information on the other 3 axes, the change on this axis will determine the new conditions that could be applied to the portfolio. The intent is to optimize the portfolio and reduce the loss. Similarly, the Time Horizon can be adjusted to reduce loss and maximize returns. For example, a longer time horizon could flatten out the intermediate changes in a stock and yield better returns. But, if the sector is not performing well and is going through a bear phase, it might actually be riskier to have a longer time horizon.

One additional step in helping the business user to make informed decisions is the evaluation of the impact of the identified conditions on the portfolio, based on previous trades. The impact is calculated as:

$$\text{Stock PL} = \text{Quantity} \times (\text{Current Price} - \text{Avg. Purchase Price}) \quad (1)$$

$$\text{Impact} = (\text{Stock PL} / \text{Net PL of Portfolio}) \times 100 \quad (2)$$

where PL = Profit/Loss

The system will discard the conditions with the least impact. Figure 5 shows the anticipated vs. actual impact, thus allowing the portfolio manager to see which of the conditions gave the maximum impact on the portfolio, based on past trades with similar conditions, vs the actual impact on the current portfolio.

Previous trades:

System suggested	Proposed impact	Applied Rules	Executed Trades	Profit/Loss	Impact on portfolio
Default	0.00	Sector score <= 5'	new Trade("General Motors", "Sell", 392.31, 07, true, "New")	-265.96	0.17
System	2.00	Portfolio share >= 5' and Current score >= 20'	new Trade("Toyota Motor", "Sell", 4200, 109, 13, true, "Rule 2 - Reduce Exposure")	-27089.50	-7.49
System	2.00	Portfolio share >= 5' and Current score >= 20'	new Trade("Toyota Motor", "Sell", 4674, 109, 13, true, "Rule 2 - Reduce Exposure")	-1500.00	1.50
System	2.00	Portfolio share >= 5' and Current score >= 20'	new Trade("Walmart", "Sell", 10011, 50, 94, true, "Rule 2 - Reduce Exposure")	-2000.00	3.00
System	2.00	Portfolio share >= 5' and Current score >= 20'	new Trade("Walmart", "Sell", 60, 50, 94, true, "Rule 2 - Reduce Exposure")	-6000.00	1.00

Figure 5. Assessing the impact of conditions using previous trades

V. GENERATING RULES

Based on the conditions identified in the previous step, the web application suggests new rules that can be edited by the business user before deploying them.

1. Suggested rules:

Name	Condition(s)	Action	Price Factor	Quantity	Anticipated Impact
New 2	Sector score < 5.00%	Sell	11%	auto/0	3%

add to rules

2. Rules editing:

Condition: Variable: Operator: Value: Add condition

Name	Condition(s)	Action	Price Factor	Quantity
New 1	Overall score > 5.00%	Sell	11%	auto/0

confirm

Figure 6. Rules editing

The information associated with a rule, as shown in Figure 6, consists of *Name*, *Conditions*, *Action*, *Price Factor*, *Quantity*, and *Anticipated Impact*. The *Name* is used to easily identify each rule among all the existing rules. The combination of one or more *Conditions* of a rule would get matched to the active data during deployment. Each condition has the form *<variable> <operator> <value>*, where the variables are facts from the domain of application. The *Action* determines what kind of trade should occur; in a stock trading scenario, there are only 2 possible actions, either “Sell” or “Buy”. In a different scenario or application, the options would differ accordingly. The *Price Factor* denotes the price at which either of the action should happen. The options for this solution are: default (11%), fixed amount (“10”), percentage (“5%”), incremental (“up 3%”). The system uses the price determination based on the *percentage stop-loss method*, where the selling price is determined based on a set percentage of the initial purchase price of the stock, and the current price. For this application, the default percentage is set to 11%, as the middle of the normal range of 5-15%, based on the risk tolerance of the investor. The options for expressing the Price Factor provide the user with extreme flexibility in creating the rule. The *Quantity* determines the number of shares to be bought or sold. The options available to the user are: as percentage, as fixed quantity, or system determined based on rules. When the quantity is set as “100%”, the entire quantity held is sold, or if the action is “buy”, the number of shares will be doubled. Whereas, if the quantity is set as “50”, a fixed quantity of 50 shares will be bought or sold. If the quantity that is available for sale is less than the denoted quantity, the existing quantity is sold. The other option is “auto/0”, this is the default option where the system determines the quantity based on the portfolio share. If the portfolio share is greater than 5%, the system will determine the necessary quantity to bring the portfolio share to 5% (normal range for each stock is 3-5% shares of the entire portfolio) [17]. Based on past trades with similar conditions,

the system calculates the *Anticipated Impact* of the rule on the portfolio, according to equations (1) and (2). For the sample set we used, the value of the anticipated impact was 3%.

Each field of a rule can be edited, with the exception of the *Anticipated Impact*. To help the business user make informed decisions when editing or creating new rules, the system will support the visualization of the stock movement for the previous 4 weeks, for each stock in the portfolio, as a trend line chart.(see Figure 7).

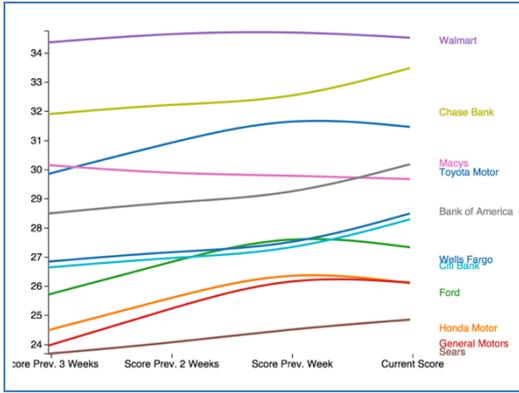


Figure 7. Stocks score trends

The stock scores in Figure 7 are calculated based on a weighted method that takes into consideration various variables for each stock, as follows:

1. Source the stock historical data - Averaged weekly
2. Source the sectorial data - Taken monthly
3. Perform sentiment analysis - Averaged Weekly
4. Calculate 50 day EMA (Exponential Moving Average) for each score (stock, sector, sentiment).

EMA stands for Exponential Moving Average, a stock price average that applies more importance to the recent prices, reacting faster to the recent price changes, than a simple moving average.

5. Generate the Overall Score for each stock in the portfolio:

$$Overall\ Score = StockEMA * 0.5 + SectorEMA*.25 + SentSc*.25 \quad (3)$$

where $StockEMA = 50$ day Stock EMA
 $SectorEMA = 50$ day Sector EMA
 $SentSc =$ Sentiment Score

The proportions for stock score, sector score and sentiment score, are defined by the portfolio manager as part of the application setup process. Higher overall scores are safe. The weights of the business parameters within a system can be adjusted by the business user, thus giving him the fine-grained control over the outcomes. In the current application the weights have been assumed as 0.5 for the Stock prices, 0.25 for Index and 0.25 for Sentiment analysis results, respectively. These weights can be different for a different application.

The calculations of scores for a particular stock (EMA, Sentiment, sector scores, overall score) are done using Map-Reduce on the data from MongoDB.

When editing rules, the system can also suggest conditions that have been used in the past, and have had a high impact on the portfolio. The best three conditions are shown as suggestions.

Before deciding on the deployment of a particular rule, the business user can visualize the effect of applying it on the portfolio. This kind of what-if scenario is visualized using a *Current vs. Target State* scatter plots, of profit/loss versus sector score (see Figure 8). The target state is plotted using the portfolio structure that would shape-up once the trade rules are executed.

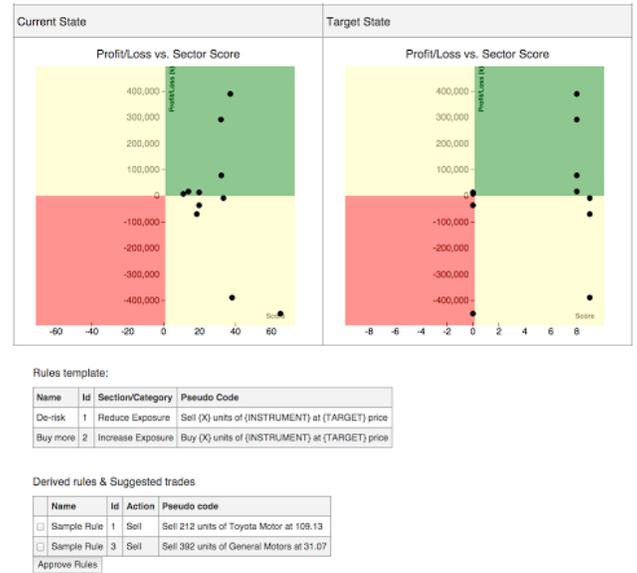


Figure 8. What-if scatter charts showing the risk of the stocks in a portfolio

A scatter chart is divided into 4 zones: left bottom quadrant is the high-risk zone, the top right quadrant is the ideal desired zone that is the safe zone, the left top and right bottom quadrants are the zones of moderate risk. The high-risk zone denotes area of loss and negative score. The safe zone denotes the area of profit and positive score. The moderate risk zones have either the Profit/Loss or the Sector Score as negative values. Stocks mapped in the high-risk zone are prone to higher risk; hence, stocks present in this zone are to be exited. The stocks present in the safe zone are the ones that would yield the best returns within the current portfolio. It would be a good idea to reduce the current holding of stocks present in a moderate risk zone, when the near term trend is negative.

All these actions help in reducing risk. The idea of the *Target State* is to move the portfolio into the top right quadrant zone, by adjusting the portfolio distribution and exposure to risk, hence resulting in minimized loss and better returns. This movement of stocks is based on the following algorithm that identifies the appropriate rules to do it:

1. Calculate a +/- range from the Overall score, based on the average movement of the sector index in the

preceding month (for example, if the sector index for Automotive had a low of 50 and a high of 70, the expected variations will be +/- 10). This range will determine the target state upper & lower limits.

2. Calculate exposure risk

$$\text{exposure risk} = \frac{\text{SectScore} * \text{AvgStockPr}}{\text{OverallVal} * 100} \quad (4)$$

where *SectScore* = Sector wise Score
AvgStockPr = Percentage of the individual average stock price for the week
OverallVal = Percentage of the overall stock value held

3. Based on the exposure risk value, retrieve stop loss rules (where price is below purchase price, or where there has been a consistent negative trend)
4. Derive new trade rules based on the stop loss rules retrieved in step 3.

VI. RULES DEPLOYMENT, EXECUTION AND FEEDBACK

For each rule in the system, the conditions are run against the data of each stock in the portfolio. If there is a match between a rule condition and a stock data, then the action part of the rule is matched against the action parts of a set of rule templates. Assuming that a rule is matched to a rule template, a new trade rule is generated, and will be placed to the stock exchange (executed in Nools). For example, if we start with the rule in Figure 9,

Sample Rule	Portfolio share >= 5%	Sell	11%	auto/D	edit
-------------	-----------------------	------	-----	--------	------

Figure 9. Sample rule

a possible match from the current portfolio, could be the Toyota Motor stock, because its portfolio share is 8.62% (as seen in Figure 10), and therefore is greater than 5%.

Name	Current Holding	Avg. Pur. Price	Current Price	Purchase Cost	Current Value	%age	Profit/Loss
				Total:	13,855,830.00	14,217,700.00	361,870.00
Toyota Motor	10,000	116.17	122.62	1,161,700.00	1,226,200.00	8.62	64,500.00

Figure 10. Snapshot of the current portfolio showing Toyota Motor

A rule template that matches the action of the Sample rule (sell), could be the Reduce Risk rule template “Sell {X} units of {INSTRUMENT} at {TARGET} price”, as in Fig. 11:

Rules template:

Name	Id	Section/Category	Pseudo Code
De-risk	1	Reduce Exposure	Sell {X} units of {INSTRUMENT} at {TARGET} price
Buy more	2	Increase Exposure	Buy {X} units of {INSTRUMENT} at {TARGET} price

Derived rules & Suggested trades

Name	Id	Action	Pseudo code
<input type="checkbox"/> Sample Rule	1	Sell	Sell 212 units of Toyota Motor at 109.13
<input type="checkbox"/> Sample Rule	3	Sell	Sell 392 units of General Motors at 31.07

Approve Rules

Figure 11. Rule template

In the template, the variables “{X}”, “{INSTRUMENT}” and “{TARGET}” are bound to specific values for *quantity*,

stock name, and *price* respectively, as follows: for quantity, as the portfolio share of Toyota Motor is 8.62, applying the rule, it should be brought down to 5% by selling the excess quantity of the shares, so to achieve that $10000 - (10000 * 5/8.62)$ results in 4200. The price is determined using the price factor set for the matched rule, therefore the current price being 122.62, applying a price factor of 11%, the resulting sell price is $122.62 - (122.62 * 11/100) = 109.13$. The stock name is bound to Toyota Motor. Hence, the trade generated will be “Sell 4200 units of Toyota Motor at 109.13”. This pseudo-code is later used to generate the actual trade order:

`newTrade("ToyotaMotor", "Sell", 4200, 109.13, true, "Sample Rule")`.

The mechanism in which the newly derived trades are deployed into the running system is handled as part of implementation. The implementation comprises validation of the new rules, adding them to the decision rules table and activating them. The decision rules table is stored in the MySQL database and loaded to the Nools rule engine for rule execution. This implementation provides the major benefit of implementing rules dynamically, without bringing down the system. Moreover, the business user is able to implement new rules much faster, without having to go through the entire IT development cycle, spanning over weeks to months.

Once a trade has been initiated, it shows up in the execution engine window, where the status of the execution engine and trades are shown (see Figure 12).

Deploy	Scheduled trades	Executed trades
6	Updated at: 27-02-2016 14:58:28	Updated at: 27-02-2016 14:58:28
	2015-12-24 05:02:08 : new Trade("Toyota Motor", "Sell", 4200, 109.13, true, "Sample Rule")	new Trade("Toyota Motor", "Sell", 4200, 109.13, true, "Sample Rule")
	2015-12-24 22:01:05 : new Trade("General Motors", "Sell", 392, 31.07, true, "New 1")	
	2015-12-24 22:00:55 : new Trade("Toyota Motor", "Sell", 158, 109.13, true, "Rule 2 - Reduce Exposure")	
	2015-12-24 22:00:55 : new Trade("Ford Motor", "Sell", 1200, 29.49, true, "New 2")	

Figure 12. Execution Engine Window

Name	Current Holding	Prev. Profit/Loss	Profit/Loss	Trade Status
		Total:	-688,138.00	361,870.00
Toyota Motor	10,000	-965,508.00	64,500.00	Queued
Honda Motor	12,000	-35,880.00	-35,880.00	
Ford	14,000	7,980.00	7,980.00	
General Motors	20,000	13,600.00	13,600.00	
Walmart	30,000	-388,800.00	-388,800.00	

Figure 13. Deployment screen showing queued trading orders

Figure 13 shows the profit/loss table with the queued trade orders. As a trade is executed, the profit/loss table of the portfolio gets updated providing up-to date status. The resulting table is shown in Figure 14.

The executed trades move to the upper left section of the execution engine window in Figure 14, while the profit and loss table shows the post execution difference of previous profit & loss vs. current profit & loss side-by-side.

The resulting data is saved into the system’s persistent storage for supporting the analytical process that is used for

producing better recommendations of rules conditions in the future.

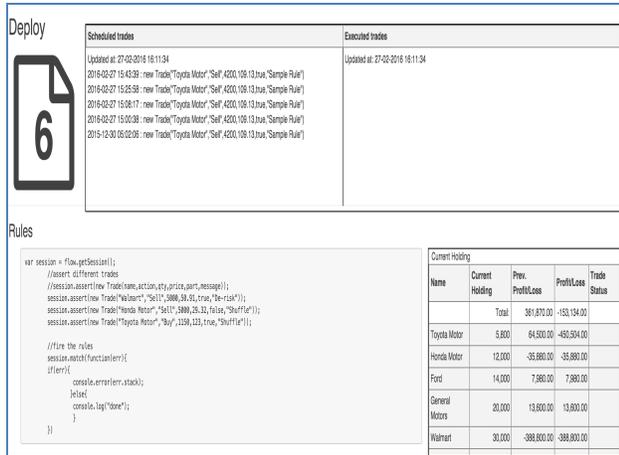


Figure 14. Executed trades and updated profit/loss table

VII. CONCLUSIONS

The approach described in this paper allows the business user to interactively and dynamically create or change stop loss business rules using data from a multitude of sources, such as historical trades, market sources, and current portfolio. Before deciding on the best rules to apply on the current portfolio, the business user can visualize the effects of applying a set of rules in what-if scenarios. Once the best rule set is identified, it is immediately deployed for execution on the active data set. The results of the execution are fed back into the system's data storage, to be used in later decision processes. The main beneficiaries of this approach are the executives who feel too dependent on the IT staff for materializing their business ideas into working enterprise systems. With this solution, a business executive is in control of the business, because of the support provided for making informed and timely decisions.

This solution addressed the problem of reducing impending loss in financial trading using open source components to accomplish the task. The ability to use different sources of information, collate heterogeneous data into a common factor, and process it to come up with decision rules, provides an inexpensive solution for businesses of any size.

This paper demonstrates the proposed solution using rules that require simple conditions. In the near future, we intend to extend this work to rules with complex conditions. The rules derived in this paper refer only to Reduce or Increase Risk Exposure rules. However, we want to extend this work with other types of stop loss rules, such as Intra-sector Shift Exposure or Inter-sector Shift Exposure. Also, we intend to extend this solution to other industries, such as retail, or healthcare, and report on the extent of work needed to adapt to new industry models. In this paper we have used only a linear

regression algorithm for processing the various types of data. We plan on experimenting with other Machine Learning algorithms, to see which one will provide the best performance for deriving business rules.

Disclaimer - The Company names and financial data are fictional, and used to solely demonstrate the proposed solution. In no way they resemble real company financial information.

REFERENCES

- [1] H. Yang, P.Liang, "Identification and Classification of Requirements from App User Reviews," *SEKE'15*,
- [2] E. Guzman, W. Maalej, "How DO Users Likes This Feature? A Fine Grained Sentiment Analysis of App Reviews", *RE'14*, p.153-162.
- [3] H. Takahashi, H. Nakagawa, T. Ysuchiya, "Towards Automatic Requirements Elicitation from Feedback Comments: Extracting Requirements Topics Using LDA", *SEKE'15*
- [4] L.V. Galvis Carreno, K. Winbladh, "Analysis of Users Comments: An Approach for Software Requirements Evolution", *ICSE'13*, p.582-591, IEEE Press
- [5] H. Dumitru, M. Gibiec, N. Hariri, J. Cleland-Huang, B.Mobasher, C. Castro-Herrera, M.Mirakhorli, "On-Demand Feature Recommendations Derived from Mining Public Product Descriptions", *ICSE'11*, p.181-.
- [6] W. Van der Aalst, "Process Mining: Discovery, Conformance, and Enhancement of Business Processes, Springer, 2011
- [7] C. Rolland, "Capturing System Intentionality with Maps", in *Conceptual Modelling in Information Systems Engineering*, p. 141-158, Springer, 2007.
- [8] G. Khodabandelou, C. Hug, C. Salinesi, "Mining Users' Intents from Logs", *International Journal of Information System Modeling and Design*, IGI Global, 2015, Special Issue from the 8th IEEE International Conference on Research Challenges in Information Science (RCIS): 2014, p. 43-71
- [9] B. von Hale, "Business Ruels Applied: Building Better Systems Using Business Rules Approach", John Wiley & Sons, New York, 2001
- [10] R.Ross, "Principles of the Business Ruels Approach", Addison-Wesley Professional, 2003
- [11] Semantics of Business Vocabulary and Business Rules (SBVR), v1.0 OMG, <http://www.omg.org/spec/SBVR/1.0/>, 2008
- [12] P. B. Feuto Njonko, S. Cardey, P. Greenfeld, W. El Abed, "RuleCNL: A Controlled Natural Language for Business Rules Specifications", Volume 8625 of the series *Lecture Notes in Computer Science*, p. 66-77
- [13] L. Nemuraite, T. Skersys, A. Sukis, E. Sinkevicius, L. Ablonskis, "VETis Tool for Efditing and Transforming SBVR Business Vocabularies and Business Rules into UML & OCL Models", 16th International Conference on Information & Software Technologies, pp. 377-384, 2010
- [14] O.C. Tantan, J. Akoka, "Automated Transformation of Business Rules into Business Processes, From SBVR to BPMN", *SEKE2014*
- [15] B. Lucie, "Report on State of the Art and Prospective Evolution of Formal Languages for Business Rules", Public Research Center Henri Tudor, Luxembourg, 2006
- [16] D. Rosca, S. Greenspan, C. Wild, "Enterprise Modeling and Decision-Support for Automating the Business Rules Lifecycle", *Automated Software Engineering*, vol.9, p.361-404, 2002
- [17] H.M. Markowitz, "Portfolio Selection". *The Journal of Finance* 7 (1): 77-91., March 1952.

TACE: A Toolkit for Analyzing Concept Evolution in Computing Curricula

Tiejian Luo, Libo Zhang, Lin Yang, Xin Chen
University of Chinese Academy of Sciences, Beijing, China

Abstract—Effective teaching and learning requires to knowing whether some concepts are needed to grasping. Our investigation for ACM Computing Curricula from 1991 to 2013 shows that the numbers of concepts are increased by 5 times. That phenomenon makes learners harder to distinguish which concept is update or not. In this paper, we develop a solution to explore the body of knowledge of computing. The proposed toolkit uses a graph model to represent the disciplinary knowledge structure. The analytic results by TACE give us insight for the various subjects in computing discipline. Our findings show that 61.3% concepts in CC1991 are obsolete. In CC2001, the proportion of obsolete concepts drops to 11.5%, and in CS2008 it is 16.8%. The OS, IM, DS, AL’s knowledge areas are more stable than CN, NC, GV, AR. The TACE’s framework is highly modular, adaptive and extendible for analyzing other discipline’s curricula.

Keywords—Teaching and Learning; Curricular Knowledge Model; Concept Vitality; Information Visualization.

I. INTRODUCTION

All subjects are built on a foundation of a unified concept family. As the field develops, new concepts are introduced, and existing concepts could change continuously due to the emergence of new knowledge. For example, the concept “big data” was invented as a result of the technological development in highly distributed and scalable systems to process massive datasets [1]. And “smart phone” nowadays refers more to mobile phone with Android and IOS rather than with Symbian OS [2]. In any discipline, to achieve good teaching and learning performance, it is essential to understand concept evolution. Here we define concept evolution as the changing of a concept and its importance in the subject. Take a computer science course example, fundamental concepts such as “data structure” and “algorithm” are playing a significant role. Some concepts gradually became important as time elapses. For instance, “Networking” and “World Wide Web” lack emphasis at the beginning of the 1990s, but they became critical foundation of computer science a decade later [3]. Some concepts become outdated after being replaced by newer concepts. A representative example is the shifting of programming language from “COBOL”, “Fortran” in the mid-1950s to “Java”, “C++”, “PHP” etc. today.

We investigated the Computing Curricula published by ACM and IEEE-Computer Society, and we found that the concepts in the curricula are used to describe CS body of knowledge have increased from 465 in to 2685. By studying the development of the curricula’s concept system, we hope to further explore the following questions:

How many concepts are there still vitality and the others are obsolete in the past 45 years? How much the proportion for the vitality and obsolete concepts from CC1991 to CS2013 curricular volumes respectively? Which knowledge areas are changed faster? Which knowledge areas are more stable? How many new knowledge units are emerged from CC1991 to CS2013?

Analyzing so many concepts in CC1991 to CS2013 curricular volumes manually is not a easy work. That motivates us to develop a toolkit termed as TACE to fulfill this task. TACE automatically extracts curricular volume’s body of knowledge, represents them in a unified graph model, it presents results with various diagrams. TACE helps us mining ACM computing curricula volumes and get some useful findings.

We organize the rest of paper as follows. In Section 2 we review related works. In Section 3 we introduce the Computing Curricula dataset. Section 4 shows how we model the knowledge structure of Computing Curricula. The framework of TACE is presented in Section 5. Section 6 describes the application of TACE and some finding for analyzing ACM Computing Curricula from 1991 to 2013. In Section 7 we give the conclusion and future work.

II. RELATED WORK

Bibliometrics methods are applied to explore research trends in different fields[4][5][6][7], which reflect the changing of topics in academic research. But research frontiers cover only part of a discipline’s knowledge. Bogoiavlenski et al. [8] reviewed the evolving computing curricula in historical perspectives. The Computing Curricula 2005 overview report [9] briefly summarized the development of the computing field before, during and after 1990s in the experts’ view. Their conclusions depict the whole picture but are not enough for understanding how disciplinary concepts evolve. Marshall [10] makes a comparison to CC2001, CS2008 and CS2013 by modeling their knowledge structure as trees. That paper discusses the knowledge unit at topic level, whose granularity is too low comparing to concept level. Moreover, it has not explored how a concept and its implication change over time, which is our research theme in this paper. Our study not only intends for computing discipline, but also tries to develop a general framework and toolkit for analyzing other disciplines’ curricula. There are considerable number of software tools for information extraction, storage and visualization. But a holistic solution should overcome problems across all these areas. A generic toolkit’s requirements and design’s specifications are summarized as follow:

- Concept annotation (CA): The toolkit should be able to

¹ CC1991, CC2001, CS2008 and CS2013 are abbreviations for Computing Curricula 1991, Computing Curricula 2001, Computer Science Curriculum 2008 and Computer Science Curriculum 2013.

² For the abbreviations of knowledge areas, see Fig. 7

extract concepts and mark tags to them, which are the basic knowledge elements in the curricula. This task's precision and recall performance should exceed 90%.

- Knowledge model representation (KMR): The system should maintain a structural data model to represent the body of knowledge in the curricular volumes.
- Knowledge model analysis(KMA): We need to develop two algorithms for dealing with the knowledge model; They will compare concepts and calculate concept's vitality according to the predefined criterion respectively.
- Data persistence function (DPF): The toolkit should provide persistent mechanism which can index and access to both data and their analytic results.
- Result export and visualization (REV): The toolkit should be able to export the results in CSV files then provide the three typical presentations: (1) statistic diagram (bar chart, pie chart and bubble chart etc.) for charting the amount, distribution or proportion of the concepts in the curricula, (2) Alluvial diagram³ for plotting the knowledge areas evolution, and(3) relationship diagram for showing the interconnection between knowledge areas.

TABLE I. RELEVANT TECHNOLOGIES

	CA	KMR	KMA	DPF	REV
Wikipedia Miner	✓	×	×	×	×
Dbpedia Spotlight	✓	×	×	×	×
GATE	✓✓	×	×	×	×
Neo4j	×	✓✓	×	✓✓	×
ggplot2	×	×	×	×	✓
MapEquation	×	×	×	×	✓
Sankey Diagram	×	×	×	×	✓

We examined several information analysis and visualization tools by comparing five key features which are matter for the curricular knowledge volumes analysis requirements. The tools we investigated are listed in table I. Wikipedia Miner [11] and Dbpedia Spotlight [12] provide automatic annotation service, but their precision and recall are below 90% [12]. GATE [13] allows us to annotate the concepts and their relationship manually, annotation performance is good. Neo4j [14] opens an API for us to build, access and store our graph model. ggplot2 [15] supports plotting basic statistic diagrams. D3.js's Sankey plugin [16] can be used to plot alluvial diagram. Map Equation [17] offers functionality for generating networked diagram. Table I shows that the available technologies have their own pro and cons in terms of the five key features. How to combine those tools's good features into our solution is not trivial.

III. COMPUTING CURRICULA DATASET

The ACM Computing Curricula⁴ is one of the most impact course guidelines for undergraduate study programs in computing field. In 1968, ACM released Curriculum 68, which is the first volume of Computing Curricula. A decade later, Curriculum 78 was published. And since 1991, when ACM

and IEEE Computer Society began working together on Computing Curricula, four volumes are released: CC1991, CC2001, CS2008 and CS2013.

TABLE II. STATISTICS OF CURRICULUM 68 AND CURRICULUM 78

Year	Course	Topic			Course Hour
		core	elective	total	
1968	22	75	119	194	71
1978	18	47	57	104	59

TABLE III. STATISTICS OF CC1991, CC2001, CS2008 AND CS2013

Year	KA	KU			Topic			Course Hour
		core	elective	total	core	Electiv	total	
1991	11	55	N/A	55	214	N/A	214	271
2001	14	63	69	132	402	548	950	280
2001	14	65	81	146	423	596	1019	290
2013	18	84	79	163	493	619	1112	307

Why do we exclude Curriculum 68's and Curriculum 78's from our dataset? The reason is that their disciplinary knowledge is organized in the exemplary courses form, which is significantly different from the later four volumes. They have three categories and are with hierarchical relationships.

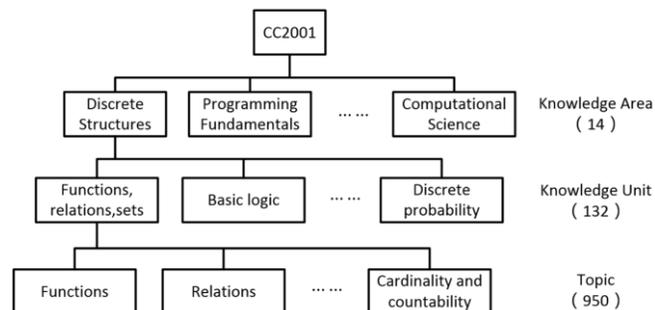


Fig. 1. Hierachy of CC2001.

The four volumes arrange the CS body of knowledge in a hierarchical way. Fig 1 shows CC2001's body of knowledge and Table III presents the statistics from CC1991 to CS2013. The whole CS field is subdivided into Knowledge Areas (KAs), which are broken down further into Knowledge Units (KUs) and correspondent topics. In CC2001 and CS2008, some of the KUs are selected as core units, which are considered essential elements for a CS undergraduate degree. The rest of the KUs are elective, which are optimal for devising a curriculum. A minimum number of hours, which should be spent on instruction, are suggested for each core KU. CC1991 and CS2013 are little different from the former two volumes. CC1991 does not use the designation for core and elective because all its KUs are considered to be essential. CS2013 moves the designation to topic level, and breaks down the core topics further into core tier 1 and core tier 2. To model all these volumes in a uniform manner, we treat all KUs of CC1991 as core. And for CS2013, we ignore the difference of the two core tiers and label all them as core.

³ http://en.wikipedia.org/wiki/Alluvial_diagram

⁴ All retrieved from <http://www.acm.org/education/curricula-recommendations>, May 12, 2014

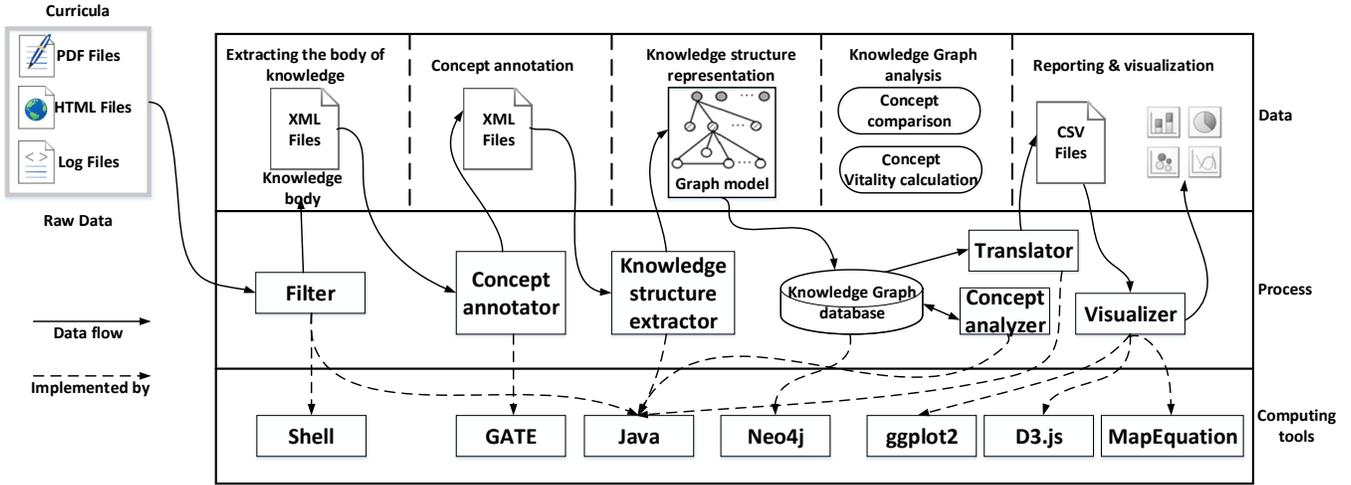


Fig. 2. The TACE framework.

IV. MODELING THE KNOWLEDGE STRUCTURE

One of the important task in our TACE framework is to develop an appropriate knowledge structure model for the curricular volumes. In order to present our approach and data model clearly, we respectively define concept, concept relation, knowledge unit, knowledge area and curriculum in this section. In addition, we define relevant concept set, which is a collection of equivalent concepts. To measure the importance level of a concept appeared in the curricula, we defined three kinds of concept vitality: steady, vitality and obsolete:

- **Steady** concepts are those that still be included in the curriculum over past two decades, such as sorting algorithm concept, “quicksort” and “mergesort” are steady.
- **Vitality** concepts have diverse weight in different periods. There are two kinds of vitality concepts. One is those appear intermittently across curricular volumes, like “scripting language”, which appears in CC2001 and CS2008, but not in CS2013. Others are concepts newly introduced in the latest curricular volume, like “Software as a service” and “Infrastructure as a service” in CS2013.
- **Obsolete** concepts are those appeared in the curricula no more than once. For example, programming languages like “Modual2” and “Ada” are obsolete, which have not been mentioned after CC1991.

V. THE TACE FRAMEWORK

TACE integrates several technologies to provide a qualitative analysis of concept evolution in the Computing Curricula. Fig 2 illustrates the overall architecture of TACE. The figure starts on the left with raw Computing Curricula raw data such as PDF files, HTML files, and Text files. The body of knowledge is extracted from them and saved in XML format. Concept annotation is subsequently employed to identify disciplinary concepts. The knowledge structure representation stage explains how we extract the annotations and build a graph model of the knowledge structure for later analysis. The graph analysis stage illustrates the analytic algorithms we

performed on the graph model. Finally the resulted information is exported as CSV files and presented in different kinds of diagrams, as described in the reporting and visualization stage.

A. Extracting the Body of Knowledge

The Computing Curricular volumes come in PDF format. We must extract the “body of knowledge” and save them in a unified structure, so that they can be processed by the component for concept annotation.

Firstly, UNIX shell program pdftotext is applied to transform the PDF files to plain text. The “body of knowledge” section needs to be picked out from the whole curricular report manually. A parser implemented by Java is responsible for parsing the section to extract KAs, KUs and topics; and save them in a unified XML format.

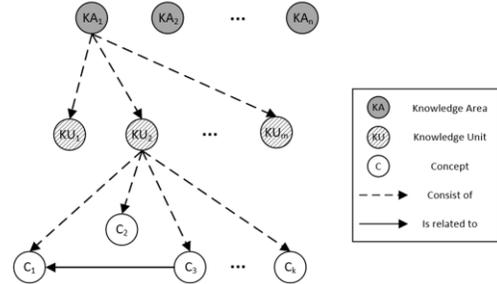


Fig. 3. Graph model of a curricular knowledge body.

B. Concept Annotation

The purpose of concept annotation is to identify the basic knowledge elements. Since the performance of publically available annotation tools are not capable enough (with best F1 score less than 60%) [12], we have to do this part work manually. We use GATE as our annotation tool. GATE takes the XML output of pre-processing stage as input. It automatically recognizes the XML tags, so we do not have to annotate KAs and KUs. The focus is to identify disciplinary concepts contained by the topics. Each concept is represented by a term. We treat all terms of topics as concept candidates, and if a candidate meets one of the following two conditions, we will take it as a concept: (1)The term is an entry of Wikipedia’s “Computer Science” Category. (2)The term is confirmed as a computer science concept by a domain expert.

The annotated result is also saved in XML format. The annotations can be extracted through the GATE’s Java API.

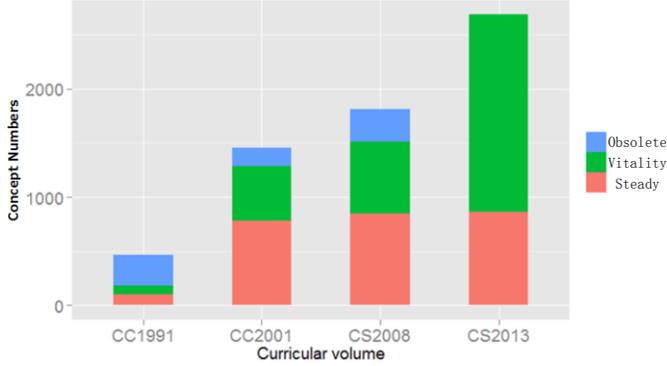


Fig. 4. Concepts progress hisgram from CC1991 to CS2013.

C. Knowledge Structure Representation

Since we have defined a linear path from KA to concepts, it can model the curricular volume’s body of knowledge as a tree-structured graph, which is shown in Fig 3. There are three kinds of vertex: KA, KU and concept. These vertices are connected by two kinds of edges: a KA vertex consists of several KUs and a KU consist of several concepts; a concept vertex may be related to other concepts that provide context for determining the exact concept’s semantics. We employ Neo4j, an open source graph database, as data storage for our graph model. Neo4j provides a native Java API, which can be seamlessly integrated with GATE’s API, allowing us to extract the model from the annotation results. Also it provides persistent and indexed access to both nodes and edges. Fig.4 gives an overview of those classes which represent the graph model of knowledge structure, along with some selected properties and methods.

D. Graph Analysis

Algorithm 1 Compare_Graph(G_1, G_2)

Input:

G_1, G_2 : graphs each representing a curricular volume

Output:

G : graph consisting of G_1, G_2 and all added edges

Procedure:

1. For each KU_A in G_1 , KU_B in G_2 do:
 - A. Compare each child C_{Ai} of KU_A to each child C_{Bj} of KU_B
 - B. If $term(C_{Ai}) = term(C_{Bj})$, add (C_{Ai}, C_{Bj}) to collection $S_{candidate}$.
 2. If $name(KU_A) = name(KU_B)$ or $|S_{candidate}| > 2$, then for each pair (C_i, C_j) in $S_{candidate}$, connect C_i, C_j with an edge SAME_AS
 3. Otherwise, for each pair (C_i, C_j) in $S_{candidate}$ do:
 - A. If $term(C_i)$ has more than 1 word, connect C_i, C_j with an edge SAME_AS
 - B. Otherwise, compare related nodes of C_i, C_j to related nodes of C_j , if there is one pair sharing a same term, then connect (C_i, C_j) with an edge SAME_AS
-

The analyzer is implemented in Java and it performs analysis on the graph model. The results are saved as properties of nodes and edges in the graph database. The analyzer’s core function is to study the vitality of concepts in each curricular volume. To determine which volumes a concept has appeared in, it is necessary to compare concepts of different volumes. To decide whether a concept is the same as another one, we made some essential definitions based on observations to the dataset. The methodology for comparing two curricular volumes is

presented in Algorithm 1. Since KU is the minimum contextual unit, two different curricular volumes are compared KU by KU. Algorithm 2 describes the procedure of identifying whether a concept is vitality or not in each curricular volume. The related definitions used in algorithms are given in Section 4.

Algorithm 2 DetermineConceptVitality (G)

Input:

G : graph consisting all the curricular volumes, and each of them has been compared to others by applying Compare_Graph(G_1, G_2)

Output:

G : the input graph with all Concept’s vitality has been determined

Procedure:

1. Set Q to contain all concept vertices:
 2. While Q is not empty, do the following:
 - A. select a vertex v in Q
 - B. Find the strongly-connected-component of v and set it to S , record the curricular volumes of S ’s vertices in Y
 - C. If Y contains CC2001, CS2008 and CS2013, for each vertex in S , set its vitality to *constant*
 - D. Otherwise, if Y contains no other than CC1991 and CC2001, or Y contains only CS2008, for each vertex in S , set its vitality to *obsolete*
 - E. Otherwise, for each vertex in S , set its vitality to *variable*
 - F. Remove all vertices of S from Q
-

E. Reporting and visualization

A result dumper is implemented to save the analytical outcome as CSV files. Meanwhile, some visualization tools are utilized to present the results in useful way. We apply ggplot2 to plot basic statistical diagrams such as bar chart and scatter chart. We use the Sankey Plugin of D3.js to demonstrate the whole picture of how different Knowledge Areas evolves. MapEquation is employed to show how knowledge areas interconnect to each other within a same body of knowledge. Some of the visualization results are displayed in Section 6.

VI. TACE APPLICATION

A. Environment Setup

All experiments are done with an Intel Core i7 PC having 8GB main memory and running Windows 7. Versions of the JRE and Neo4j used are: 1.7.0 45, 2.0.0 respectively. The ggplot2 package version is 0.9.3.1 and the R version is 3.0.3.

B. Approach

The workflow of TACE follows the sequence presented in Section 5. The dataset is introduced in Section 3. We utilized pdftotext to transform the 4 raw PDF files to plain text and manually selected out the sections of “body of knowledge”. Then we employed the parser to parse them and save the output as XML files. We loaded them with GATE for concept annotation. It takes a volunteer student almost 38 hours to fulfilled concepts annotation work. It includes 6416 concepts and 363 concept relations. The final annotated concepts are saved into 4 XML files. We invoked the controller program of TACE and it automatically finished the process of knowledge structure representation and analysis. We wrote an R script for applying ggplot2 and an Excel VBA script for generating Sankey Diagram. MapEquation was utilized to plot KA interconnection diagrams.

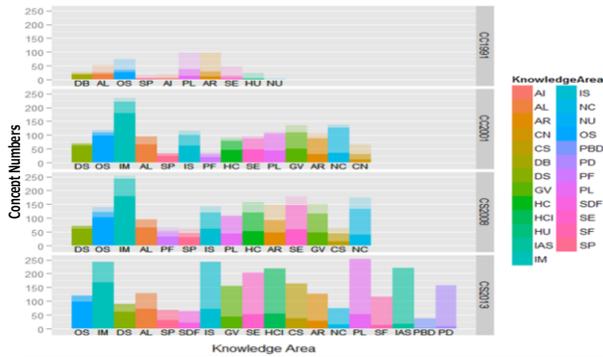


Fig. 5. Concepts distributions from CC1991 to CS2013.

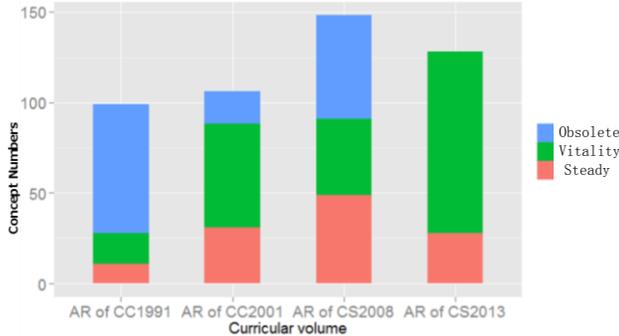


Fig. 6. Concepts variation histogram from CC1991 to CS2013 for "Architecture and Organization" knowledge area

C. Analytic Results

We got four kinds of result set as following:

- **Concept distribution:** The counts of concept in KA and KU for each curricular volume. For example, CS2001 contains 1452 concepts. Among which, there are 119 concepts in "Operating System" KA and there are 10 concepts in the "Concurrency" KU.
- **Concept vitality:** It has been defined in Section 4. For example, the concept "floating-point" is steady while "ada" (programming language) is obsolete.
- **Concept origin:** It means that a concept is either newly introduced or inherited from a former curricular volume. For instance, the concept "Bayes theorem" is newly introduced in CC2001 and inherited by CS2008 and CS2013.
- **Concept duplication:** It stands for those concepts have occurred repeatedly in most curricular volumes. For example, the concept "predicate logic" appeared in both "Knowledge Based Reasoning" KU and "Basic Logic" KU.

As space is limited, we only present some key analysis results in this paper. Fig 4 shows that CS discipline has been developed and progressed rapidly. From the perspective of concept vitality, we may find most concepts of CC1991 become obsolete, while in CC2001 and CS2008 obsolete concepts only take a small part. The number of new concepts in CS2013 accounts for most vitality concepts.

From the steady concepts perspective in Fig 5, we may find most topics in DS, OS, IM, and AL are always considered important for undergraduate curriculum. This is in accordance

with their fundamentality in CS. From the obsolete concepts perspective, most of the CN's concepts are no longer

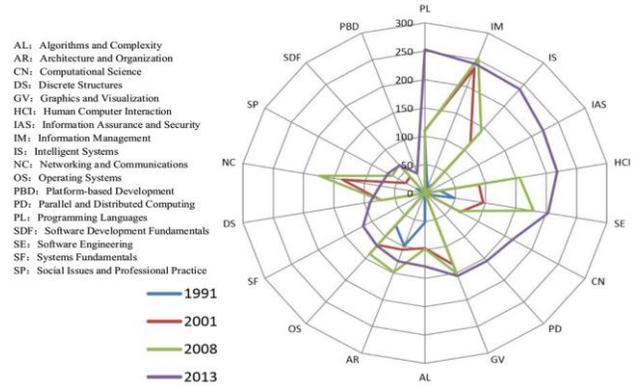


Fig. 7. Concepts radar graph from CC1991 to CS2013.

considered as necessary. This is because CN has a relatively small KA and contains no core KUs. Its content varies across the volumes and results in the large proportion of obsolete concepts in CS2001.

Specifically, we thoroughly observed at the concept vitality of "Architecture and Organization" from CC1991 to CS2013 in Fig 6. The count of concepts kept increasing in first three volumes but shrank at CS2013. The large proportion of obsolete concepts in CS2008 contributed to the decrease. Another reason for the drop was that some concepts in AR were categorized to other KAs like SF (System Fundamentals) and PD (Parallel and Distributed Computing). This also explained the drop of steady concepts' percentage in CS2008.

In order to deeply understand the change of Knowledge Area in each curricular volumes, we drew the radar chart based on the distribution of concepts' number in the Knowledge Area, as shown in Fig 7. We have observed that the content of Knowledge Area has been increasing year by year. Some areas always have an important role, such as IM, SE and GV. Some new areas appear like IAS, SF and some like NC have been outdated. The change of human knowledge is like the evolution of Fig 7. Although the importance of each field is fluctuating, in the overall trend, all Knowledge Area has been improving gradually.

In Fig 8 the four vertical columns blocks represent the topics of different curricula (1991,2001,2008,2013), the height of each block stands for the weights of this topic in its curricula. The lines between different vertical columns represent the mapping of topics in two curricula (Thicker the line is, more part of the topic in the left column transfer into the topic in the right column). For instance, the content of topic "Artificial Intelligence and Robotics" in 1991 curricula transfer into "Algorithm and Complexity", "Human-Computer Interaction" and "Intelligent Systems" in 2001 curricula. The new topic "Algorithm and Complexity" in 2001 curricula assimilate the information of 4 topics in 1991 curricula. Topic "Algorithm and Data Structure" and topic "Programming Languages" provide larger proportion of content than topic "Artificial Intelligence and Robotic" and "Architecture". At last, topic "Algorithm and Complexity" is composed of the transfer content and the new content.

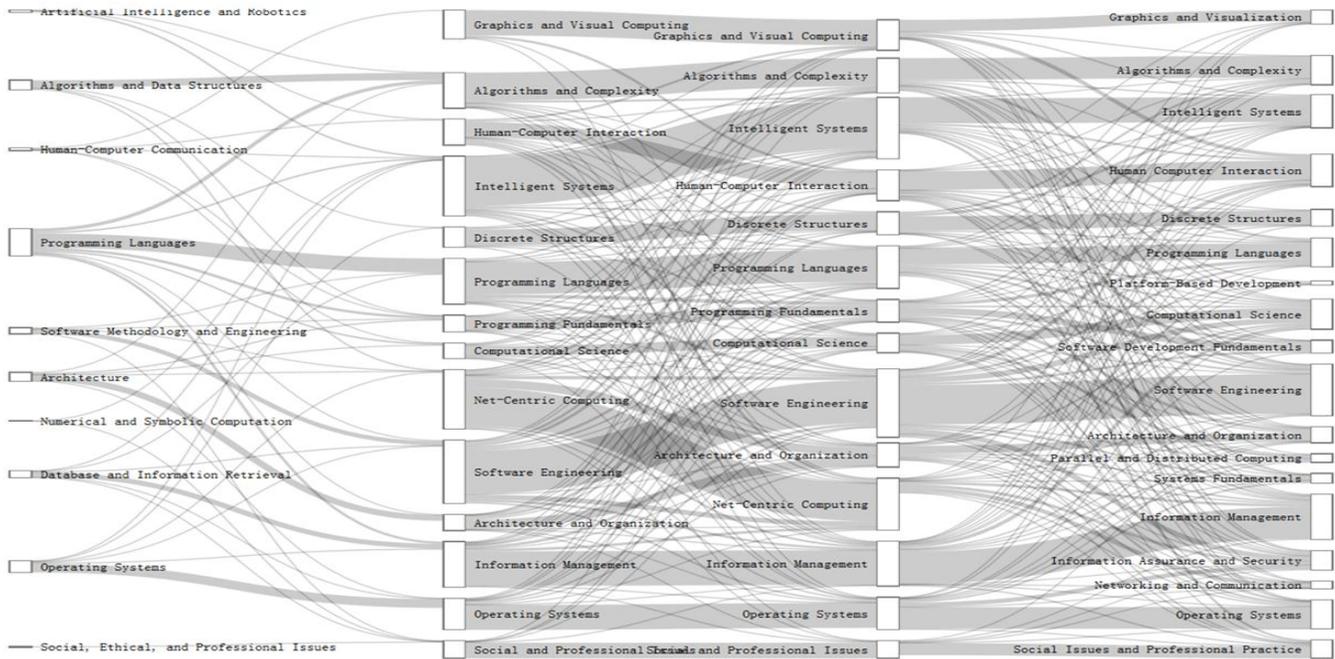


Fig. 8. New knowledge units are emerged from CC1991 to CS2013.

VII. CONCLUSION

In this paper, we presented TACE, a novel toolkit that enables people to explore the concept family's development of computer science thoroughly. Our major work is four folds. First, we investigated the Computing Curricula, which is the dataset of TACE. Second, we defined a concept vitality metrics and introduced a graph model to represent the curricula's body of knowledge. Third, we presented TACE's framework in 3 layers: business workflow, software architecture and implementation tools. In detail, we described the 5 phases of the business workflow. Finally, we demonstrated some key findings of TACE, which illustrate the Computing Curricula's development in the aspect of concept's vitality. Given the similar knowledge hierarchy of all scientific disciplines, we hope its methodology can be applied to other subjects.

Our study shows that the number of knowledge areas has doubled and the amount of concepts has increased by 5 times from CC1991 to CS2013. The 61.3% of concepts in CC1991 are obsolete. In CC2001, the proportion of obsolete concepts drops to 11.5%, and in CS2008, it is 16.8%. From the steady concepts perspective, OS, IM, DS, AL are the most stable knowledge areas while CN, NC, GV, AR are changing rapidly. Currently the precision of analyses relies heavily on manual concept annotation, which is time-consuming and fallible. To apply TACE on a larger dataset and a broader variety of data source, we must develop a component to automatically extract domain concept with high performance.

REFERENCES

- [1] Silva Y N, Dietrich S W, Reed J M, et al. Integrating big data into the computing curricula[C]//Proceedings of the 45th ACM technical symposium on Computer science education. ACM, 2014: 139-144.
- [2] M. Butler. Android: Changing the mobile landscape. *Pervasive Computing, IEEE*, 10(1):4-7, 2011.
- [3] G. Engel and E. Roberts. Computing curricula 2001 computer science. IEEE-CS, ACM. Final Report, 2001.
- [4] K. W. Boyack, K. B ö rner, and R. Klavans. Mapping the structure and evolution of chemistry research. *Scientometrics*, 79(1):45-60, 2009.
- [5] A. Hoonlor, B. K. Szymanski, and M. J. Zaki. Trends in computer science research. *Communications of the ACM*, 56(10):74-83, 2013.
- [6] J.-I. Hung. Trends of e-learning research from 2000 to 2008: Use of text mining and bibliometrics. *British Journal of Educational Technology*, 43(1):5-16, 2012.
- [7] J. Li, M.-H. Wang, and Y.-S. Ho. Trends in research on global climate change: A science citation index expanded-based analysis. *Global and Planetary Change*, 77(1):13-20, 2011.
- [8] I. A. Bogoiavlenski, A. G. Clear, G. Davies, H. Flack, J. P. Myers, R. Rasala, M. Chairman-Goldweber, and J. Chairman-Impagliazzo. Historical perspectives on the computing curriculum. *ACM SIGCUE Outlook*, 25(4):94-111, 1997.
- [9] R. Shackelford, A. McGettrick, R. Sloan, H. Topi, G. Davies, R. Kamali, J. Cross, J. Impagliazzo, R. LeBlanc, and B. Lunt. Computing curricula 2005: The overview report. *ACM SIGCSE Bulletin*, 38(1):456-457, 2006.
- [10] L. Marshall. A comparison of the core aspects of the ACM/IEEE computer science curriculum 2013 strawman report with the specified core of CC2001 and CS2008 review. In *Proceedings of Second Computer Science Education Research Conference*, pages 29-34. ACM, 2012.
- [11] D. Milne and I. H. Witten. An open-source toolkit for mining wikipedia. *Artificial Intelligence*, 194:222-239, Jan. 2013.
- [12] P. N. Mendes, M. Jakob, A. Garc'ia-Silva, and C. Bizer. DBpedia spotlight: shedding light on the web of documents. In *Proceedings of the 7th International Conference on Semantic Systems*, pages 1-8. ACM, 2011.
- [13] H. Cunningham, D. Maynard, and K. Bontcheva. *Text processing with gate*. Gateway Press CA, 2011.
- [14] Neo4j. Home. <http://www.neo4j.org>. Retrieved June 7, 2014.
- [15] H. Wickham. *ggplot2: elegant graphics for data analysis*. Springer, 2009.
- [16] M. Bostock. Sankey plugin of d3. <https://github.com/d3/d3-plugins/tree/master/sankey>. Retrieved June 7, 2014.
- [17] D. Edler and M. Rosvall. The map generator software package. Online at <http://www.mapequation.org>, 2010. Retrieved June 7, 2014.

Sentiment Analysis of Name Entity for Text

Xinzhi Wang
Department of Engineering Physics
Tsinghua University
Beijing, China
wxz15@mails.tsinghua.edu.cn,

Jiayue Wang
Department of Engineering Physics
Tsinghua University
Beijing, China
wangjy15@mails.tsinghua.edu.cn,

Hui Zhang
Department of Engineering Physics
Tsinghua University
Beijing, China
zhui@mail.tsinghua.edu.cn,

Yang Zhou
Department of Engineering Physics
Tsinghua University
Beijing, China
zhou-y15@mails.tsinghua.edu.cn

Abstract—Recent years, big data has attracted increasing interest. Sentiment analysis from microblog as one kind of big data also receive great attention. Some recent research works are not suitable for sentiment analysis as the result that users prefer to express their feelings in individual ways. In this paper, a framework is proposed to calculate sentiment for aspects of event. Based on some state of art technologies, we build up one flowchart to get sentiment for aspects of event. During the process, name entities with the same meaning are clustered and sentiment carrier are filtered. In this way sentiment can be got even user express feeling for the same object with different words.

Keywords-sentiment analysis framework, event analysis, microblog, big data

I. INTRODUCTION

Recently, one tends to express sentiments at anytime and anywhere with the prevalence of online social media. Sina (<http://weibo.com>), Tencent (<http://blog.qq.com/>) in China and Twitter (www.twitter.com), Facebook (www.facebook.com) in USA, spread millions of personal posts on a daily basis by millions of users. Rich and enormous volume of data propagated through social media attracted enormous attention from researchers. However, information of these platforms is sparse and various, as user have different characteristics in expressing their opinions. When users describe the same objective, they may use different words. For instance some prefer using 'typhoon', and others prefer using the name of typhoon 'Rammasun' or 'Matmo' when they discuss the same event at the same time. As a result, analyzing and managing sentiment for text has become one kind of meaningful and challenging tasks in the area of affective computing, especially in microblog.

This paper propose one practical way to get the sentiment of aspects of event even they are expressed in different ways. In this paper, introduction and related work are given in section I and section II. The framework is introduced in section III.

Technologies about prominent word extraction and prominent name entity extraction are discussed in section IV. Methods of sentiment analysis for prominent name entity and experiment are denoted in section V and section VI, respectively. Finally, conclusions are made in the last section.

II. RELATED WORK

Sentiment analysis is subset of semantic analysis. Semantic analysis in processing microblogs[1] and news[2] include detecting and tracing event. Semantic information is also employed in some video processing[3], [4]. Sentiment analysis reuse the technologies semantic analysis, and focus on three levels: word level, sentence level, and article level.

Opinion words are extracted mainly through three ways. 1) manual approach, 2) dictionary based approach, and 3) corpus based approach. Manual approach is very time consuming but accurate [12]. The dictionary based approach and opinion words collection have a major shortcoming, since the same word may have different sentiment orientation in different domains. So far, several opinion word lists have been generated [13]. Semantic associations in text are mined from different aspect[14], [15]. These relations have been used in large scale news analysis[16]. Rules or constraints are designed for connectives, such as AND, OR, BUT, EITHEROR, and NEITHERNOR [17]. The idea of intra-sentential and inter-sentential sentiment consistency were explored in [18]. Qiu G[19] employed dependency grammar to describe relations for double propagation between features and opinions. Liu [20] adopted the same context definition but used it for sentiment analysis of comparative sentences. Breck [21] went further to study the problem of extracting any opinion expressions, which might have any number of words. The Conditional Random Fields (CRF) method [22] was used as the sequence learning technique for extraction. Most of above works are based on probability, which cannot express two strong emotions at the same time.

Machine learning methods are widely used in both sentence and article level. Zhu Y [26] employed naive Bayesian classi-

fier to discriminate objective and subjectivity classification. Pang [27] employed three machine learning methods (i.e., Naive Bayes, maximum entropy classification, and support vector machines) on sentiment classification for text. Subsequent research also used other machine learning algorithms such as pruning and pattern recognition methods [28]. Wilson [29] pointed out that a single sentence may contain multiple opinions. Automatic sentiment classification was presented to classify the clauses of sentiment of every sentence down to four levels (i.e., neutral, low, medium, and high) by the strength of the opinions being expressed in individual clauses [30]. So, lots of researchers tried to break down the word boundary of sentiment carrier.

All of the previous works focus on the methods of deterministic algorithm, which contribute to the development of sentiment analysis. But few have combine these effective method. This paper make efforts to get the goal.

III. TASK DESCRIPTION

With the proliferation of mass media, interest in mining sentiment and opinions in text has attracted more and more attention. Sentiment or opinions expressed by authors is affected by event, products, aspect of event, or aspect of products. Such as the sentence 'The camera of my phone is dim' expresses negative sentiment about aspect 'camera' of product 'phone'. Also the sentence 'Things went well, but the results were heartbreaking' expresses positive sentiment about event 'Things' and negative sentiment about aspect 'result' of event 'Things'.

This paper focus on the problem of finding sentiment orientation for prominent word of text. To do this job, we divide the whole process into four parts. Firstly, Extracting Prominent Word. Extracting prominent word is the basic work, which include word segmentation, removing stop words, recognizing name entity. Secondly, Extracting Prominent Name Entity. Extracting prominent name entity is based on the fact that prominent word has been mined. This part contains representation of word, computing weight of word and name entity filtration. Thirdly, Extracting Sentiment Carriers For Name Entity. Sentiment carrier can be any type of word including nouns, verbs, adjectives, and adverbs, but not all word can express opinion. In this way, we design one method to recognize sentiment word and build relationship between these carriers and name entities, and details be discussed later. Fourthly, Sentiment Analysis For Text. Sentiment carriers are recognized as they express some sentiment. In this part, which kind of sentiment is expressed is analysed. Furthermore, sentiment orientation for name entity and sentiment orientation for text will be calculated.

Every part of the whole process shown in Fig.1 has been attracting a lot of researchers' attention. This paper aims to do sentiment analysis employing improved state-of-art methods, and more details will be discussed later.

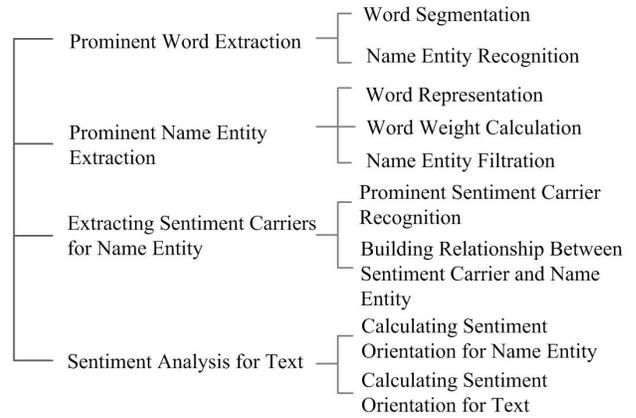


Fig. 1. Technological process of sentiment analysis for text.

IV. PROMINENT WORD EXTRACTION

To extract prominent word from Chinese, four main steps have to be considered. The three steps are word segmentation, name entity recognition, word representation, word weight calculation. With each part has been researched deeply, we choose some efficient method to finish our work. Details are discussed later.

A. Word Segmentation

Now, segmentation of Chinese word are mainly focus on three field including word based generative model, character based generative model, and dictionary based model. This paper employ one kind of word based generative model. Then the following three steps are token.

- 1) Search candidate by matching items in the word dictionary and build up wordnet like $\{(a,ab),(b,bcd),(c),(d)\}$
- 2) Get best candidate sequence employing viterbi algorithm using bigram dictionary.
- 3) Determine pos sequence using word dictionary and transfer matrix table.

MM(Markov Model) and HMM(Hidden Markov Model) are used to do segment of Chinese. In this way, word and corresponding pos can be got.

B. Name Entity Recognition

Name entity denotes name for certain specific person, location, group, or organization containing person name, location name, and organization name. Also, there are lots researches have been done, here we employ method proposed by Huaping Zhang[11], in which cascaded hidden Markov model is used for Chinese named entity recognition.

The whole process is divided into three levels, person name recognition, location recognition, group and organization name recognition. In each level, roles are employed. Furthermore, state transmission matrix and state observation emission matrix are counted. At last, patterns of roles are set to finalize which consecutive roles sequence combined to these entity name. Details can be found in [11].

After recognizing name entity, it turns to extracting prominent name entity. To achieve this goal, weight calculation method are introduced after word representation using word vector. At last, ways to filter prominent name entity are described.

C. Word Representation

One famous text representation model is VSM(Vector Space Model), in which every document is represented by a vector with every element be the IF/IDF weight of corresponding word. Also, word have always been regarded as a vector in some topic models such as LDA [7]. Here we employ GloVe [5], [6] model which has perfect performance in capturing fine-grained semantic and syntactic regularities proposed by Richard Socher.

More details can be found in the paper of Socher et al.[6]. Word vector employ one kind of unsupervised model learning word representations which have great performance on word analogy, word similarity, and named entity recognition tasks. In this way, each word was represented by a vector.

D. Word Weight Calculation

Many methods have been proposed to calculate word weight in the research domain of keyword extraction, such as classical TextRank [8] and TFIDF [9]. Both methods have been widely used in natural language processing area. TextRank functions as PageRank which is employed by Google in information retrieval. Here we employ TextRank in calculating word weight. Also, this method is often been used in abstract generation. Calculation of word weight consists of the following main steps:

- 1) Identify relations that connect vertexes, and use these relations to draw directed and weighted edges between vertexes. Weight of words relations are got as,

$$S(w_i, w_j) = ne(w_j, w_i) \quad (1)$$

where $S(w_i, w_j)$ means the frequency of both w_i and w_j appear in the same words window. Normally, the length of window is set to be five.

- 2) Calculate word weight through iteration equation as,

$$WS(w_i) = (1 - d) + d * \sum_{w_j} \frac{S(w_j, w_i)}{\sum_{w_k} S(w_j, w_k)} WS(w_j) \quad (2)$$

where d is a damping factor that can be set between 0 and 1. The damping factor is often set to 0.85. In this way, words with high weight can be easily found.

- 3) Sort vertices based on their final score. Important words are assigned high weight.

Until now, we can get prominent word after segmenting, representing and weighting words, which facilitate following calculation including name entity extraction and sentiment orientation analysis.

E. Name Entity Filtration

Name entity filtration aims to find primary name entity from large scale of text. We have introduced textrank to calculate word weight above, which means we can set one threshold to get rid of the noises. Traditionally, two ways can be employed, namely set the minimum weight or set the qualified percentage. Which method should be chosen based on specific text. More details will be discussed in experiment.

Some of these entities may be related to specific sentiments. In next section, we will discuss one method to calculate sentiment orientation of words.

V. SENTIMENT ANALYSIS FOR PROMINENT NAME ENTITY

There are both objective and subjective sentences in text, such as sentence 'MacPhee, who is a girl, would be happy to eat it!' is one compounding sentence. 'MacPhee' and 'happy' are subjective when expressing some sentiment. This section aims to mine sentiment expressed by 'happy' for 'MacPhee'.

A. Prominent Sentiment Carrier Recognition

Generally speaking, some words with specific pos (part of speech), such as adjective element, adjective, adverb element, adverb, verb, exclamation, and even some punctuation can express various kind of sentiment explicitly.

In this paper, to recognize sentiment carrier, we choose a small kernel set of 1166 terms manually as sentiment seed based on [23]. There are 6 kinds of primary sentiment including love, joy, surprise, anger, sadness, and fear. Complex emotions are various combination of the six primary sentiments. Furthermore, the six kinds of sentiment words are expanded using bootstrapping method according to one Chinese lexicon named 'HaGongDaCiLin', most of which are adjectives and adverbs. Vectors of these words are described as v_k^j , namely the k th seed word in sentiment the j th sentiment. Moreover, similarity of two words s_{v_i, v_k^j} can be got as follow:

$$s_{v_i, v_k^j} = \frac{v_i * v_k^j}{|v_i| |v_k^j|}, \quad (3)$$

where v_i is the word vector of w_i from V.A. If $s_{v_i, v_k^j} > 0.7$, then we select v_i as sentiment carrier, but we can not determine which kind of sentiment v_i express as the result that some antonyms can be used in the same situation with similar word vectors. K-means is employed when it comes to clustering.

As microblog is some kind of short text, the sentiment of these short messages keeps unanimous at most situation. Then sentiment carriers which appear in the same microblog express similar sentiment.

B. Building Relationship Between Sentiment Carrier and Name Entity

Sentiment expressed by carriers belongs to some entity, formulated as $w_i \in N(w_j)$ with $N(w_j)$ be the set of sentiment carrier for word w_j , such as word 'happy' attribute to 'MacPhee', namely ' $happy' \in N('Macphee')$ ', in sentence 'MacPhee would be happy to eat it!'. The principle of proximity within the frame of punctuation is the main criterion when

it comes to consider building relationship between sentiment carrier and name entity.

C. Calculating Sentiment Orientation for Name Entity

Actually, word weight can be got using method introduced above, which means each sentiment carrier and name entity have their values. Furthermore, sentiment carrier contribute to the sentiment of name entity in different degree. So here we using the simple average weight to deal the problem of calculating sentiment orientation for name entity.

$$se_{ij} = \frac{\sum_k WS(w_k) * se_{kj}}{\sum_k WS(w_k)}, \quad w_k \in Ne(w_i) \quad (4)$$

where $WS(w_k)$ is the weight of word w_k , and se_{kj} is the sentiment orientation of word w_k to the j th sentiment. At the same time w_k must be element of sentiment carrier set for word w_i . In this way, the prominent sentiment of name entity depend more on distinctive high weight sentiment carriers.

D. Calculating Sentiment Orientation for Text

Text is sequence of words, including name entities and sentiment carriers. Name entities act as the major descriptive content, while sentiment carriers are more like attributes for name entity. Name entity weight differently when they appear in the text. If the most weighted object is very 'joy', then the sentiment of text tend to be 'joy'. In this way, we employ one way similar to calculation of name entity to calculate sentiment orientation for text.

$$\begin{aligned} tse_{ij} &= \frac{\sum_i WS(w_i) * se_{ij}}{\sum_i WS(w_i)}, \\ &= \frac{\sum_i \sum_k WS(w_i) WS(w_k) * se_{kj}}{\sum_k WS(w_k) \sum_i WS(w_i)} \quad (5) \\ w_k &\in Ne(w_i) \\ w_k &\in Ne(te_i) \end{aligned}$$

where $Ne(te_i)$ means the name entity set appear in text te_i . tse_{ij} is the sentiment orientation in dimension j th for corresponding text. Other variables keep the same meaning as before.

VI. EXPERIMENTS

In this part, we will explain how these methods works in details. Three parts, prominent word and name entity extraction, sentiment carriers recognition, sentiment analysis for text, are included.

A. Prominent Word and Name Entity Extraction

The corpus we employed in this paper is crawled from 'weibo.sina.com' with keyword 'rainstrom' in Chinese with 339541 microblogs involved. 'weibo.sina.com' is one of the most popular social media platform just like Twitter in USA, which allows users to express opinions freely. As a result, there are large amount of noisy information mixed in the interesting contents. So, before we take steps do future study, textrank with adaptive window size(one sentence as window) is used firstly to get topic sentences while removing noisy sentences.

Details are shown in Table.I. Precision, Recall and F1-Value should be higher than 90% according to [11]. Then textrank with window size five is employed to calculate word weight.

name of items	token number
number of microblog in corpus	339541
number of word in corpus	76364
number of person name in corpus	8917
number of place name in corpus	3766
number of organization name in corpus	78
number of topic sentence from corpus	33907
number of word from topic sentences	8581
number of person name from topic sentences	545
number of place name from topic sentences	670
number of organization name from topic sentences	17

TABLE I
CORPUS INFORMATION WHEN EXTRACTING WORD AND NAME ENTITY.

Name entities with high weight are left to do future analysis. Some nouns such as 'rainstorm' and 'rain' who carries high weight are also kept for future analysis.

B. Extracting Sentiment Carriers for Name Entity

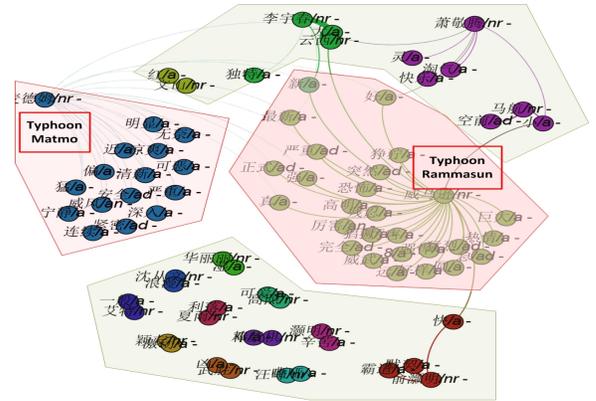


Fig. 2. Name entity (typhoon Matmo and Rammasun) and its adjectives.

As is discussed in V.A, each word is expressed by one word vector. Words with similar attributes are assigned analogous vector. Such as 'downpour' and 'rainstorm' are often decorated by 'sudden' and 'abrupt'. As a result, vector of 'downpour' and vector of 'rainstorm' is quite similar. But when it turns to adjectives, it is not that effective in finding synonyms. Distance between 'beautiful' and 'ugly' are small, as they can be used in the position such as 'the dress is ugly/beautiful'. In our practice, we manually select 1066 sentiment seeds to extract adjectives who carry sentiment. Then based on the hypothesis that sentiment stay consistent in the same microblog without negative words, we compute the sentiment orientation of these sentiment carriers. Fig.2 shows part of the results.

C. Sentiment Analysis for Text

The same object can be expressed in different ways. Such as typhoon can be express by specific typhoon name, rain can be expressed by downpour and so on. So in this part,

we cluster words with the same meaning together before calculating sentiment for them.

Then by employing sentiment carriers for cluster of name entities, we can get the sentiment orientation of aspect from text. The result show that 'fear' is strong for 'rainstorm' and 'typhoon'. Slight joy with 0.24 exists in text for 'rainstorm'. 'joy' and 'anger' both with 0.39 exist in text for 'typhoon'. Sentiment 'joy' originate from jokes produced when 'rainstorm' and 'typhoon' happen, while 'anger' and 'fear' result form their destructiveness. Follow the same steps, we can get sentiment for other entities appears in the text.

VII. CONCLUSION

This paper propose one practical framework in determining sentiment analysis of name entities. When we try to get short but meaningful information from big data, this framework can be used as guidance. In all, this paper have four contributions:

1. Propose one framework of new technologies. Large amount of new technologies have been spring up these years. Each of them have advantages and disadvantages. This paper combine some practical works such as word vector representation, textrank to get sentiment orientation for text.

2. Name entity clusters. By employing word vector representation, we find that noun words with similar meaning can be clustered easily, but when it comes to adjectives, it will not work that well. This is because, adjectives with different meaning may share the same context.

3. Recognize sentiment carriers and calculate sentiment orientation for name entities. Most sentiment is carried by adjectives. By calculating their weight, prominent word can be filtered. Then sentiment seeds are used to get the sentiment orientation after building the relation of sentiment carriers and name entities.

There are still some future works needed to be completed. Such as finding practical ways to cluster adjectives which have similar meaning. Also, more experiments should be implemented to verify the performance of related technologies in the framework.

REFERENCES

- [1] Xu Z, Liu Y, Xuan J, et al. Crowdsourcing based social media data analysis of urban emergency events[J]. *Multimedia Tools & Applications*, 2015:1-18.
- [2] Xu Z, Liu Y, Yen N Y, et al. Crowdsourcing based Description of Urban Emergency Events using Social Media Big Data[J]. *IEEE Transactions on Cloud Computing*, 2016:1-1.
- [3] Xu Z, Liu Y, Mei L, et al. Semantic based representing and organizing surveillance big data using video structural description technology[J]. *Journal of Systems & Software*, 2015, 102(C):217-225.
- [4] Xu Z, Mei L, Liu Y, et al. Semantic enhanced cloud environment for surveillance data management using video structural description[J]. *Computing*, 2014:1-20.
- [5] Huang E H, Socher R, Manning C D, et al. Improving word representations via global context and multiple word prototypes[J]. In *Proc. of the Annual Meeting of the Association for Computational Linguistics (ACL)*, 2012, 1:873-882.
- [6] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. Glove: Global vectors for word representation. In *Proceedings of EMNLP*. 2014.
- [7] Jordan M I, Blei D M, Ng A Y. Latent Dirichlet Allocation[J]. *Journal of Machine Learning Research*, 2003, 3:465-473.
- [8] Tarau P, Mihalcea R. TextRank: Bringing Order into Texts[J]. *Unt Scholarly Works*, 2004.
- [9] Salton G, Buckley C. Term-weighting approaches in automatic text retrieval[J]. *Information Processing & Management An International Journal*, 1988, 24(5):513-523.
- [10] Wu H C, Luk R W P, Wong K F, et al. Interpreting TF-IDF term weights as making relevance decisions[J]. *Acem Transactions on Information Systems*, 2008, 26(3):55-59.
- [11] Yu H K, Zhang H P, Liu Q, et al. Chinese Named Entity Recognition Based on cascaded hidden Markov model[J]. *Journal of communication*, 2006, 2:87-94.
- [12] S. R. Das and M. Y. Chen. Yahoo! for Amazon: Sentiment extraction from small talk on the Web, *Management Science*, vol. 53, pp. 1375C1388, 2007.
- [13] Rao Y, Lei J, Liu W, et al. Building emotional dictionary for sentiment analysis of online news[J]. *World Wide Web-internet and Web Information Systems*, 2014, 17(4):723-742.
- [14] Luo X, Xu Z, Yu J, et al. Building Association Link Network for Semantic Link on Web Resources[J]. *IEEE Transactions on Automation Science & Engineering*, 2011, 8(3):482-494.
- [15] Hu C, Xu Z, Liu Y, et al. Semantic Link Network-Based Model for Organizing Multimedia Big Data[J]. *IEEE Transactions on Emerging Topics in Computing*, 2014, 2(3):376-387.
- [16] Xu Z, Wei X, Luo X, et al. Knowle: A semantic link network based system for organizing large scale online news events[J]. *Future Generation Computer Systems*, 2015, s 43V44:40-50.
- [17] V. Hatzivassiloglou and K. McKeown. Predicting the semantic orientation of adjectives, *Proceedings of the Joint ACL/EACL Conference*, pp. 174C11, 1997.
- [18] S. Huang, Z. Niu, C. Shi. Automatic construction of domain-specific sentiment lexicon based on constrained label propagation[J]. *Knowledge-Based Systems*, 2014, 56(3):191C200.
- [19] G. Qiu, B. Liu, J. Bu, et al. Expanding domain sentiment lexicon through double propagation[C]// *Proceedings of the 21st international joint conference on Artificial intelligence* Morgan Kaufmann Publishers Inc., 2009:1199-1204.
- [20] G. Ganapathibhotla and B. Liu. Identifying Preferred Entities in Comparative Sentences, *Proceedings of the International Conference on Computational Linguistics, COLING*, 2008.
- [21] E. Breck, Y. Choi, and C. Cardie. Identifying expressions of opinion in context, *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, 2007.
- [22] M. Zheng, Z. Lei, X. Liao, et al. Identify Sentiment-Objects from Chinese Sentences Based on Cascaded Conditional Random Fields[J]. *Journal of Chinese Information Processing*, 2013, 27(3):69-76.
- [23] Parrott W. *Emotions in social psychology: Essential readings*[M]. Psychology Press, 2001.
- [24] Jijian Xie, Chengping Liu. *Fuzzy mathematics method and application* [M]. Huazhong university of science and technology press, 2000.
- [25] Salton G, Wong A, Yang C S. A vector space model for automatic indexing[J]. *Communications of the ACM*, 1975, 18(11): 613-620.
- [26] Zhu Y, Tian H, Ma J, et al. An Integrated Method for Micro-blog Subjective Sentence Identification Based on Three-Way Decisions and Naive Bayes[M]// *Rough Sets and Knowledge Technology* Springer International Publishing, 2014:844-855.
- [27] Pang B, Lee L. Opinion Mining and Sentiment Analysis[J]. *Foundations and Trends in Information Retrieval*, 2008, 2(1):459-526.
- [28] Karamibekr M, Ghorbani A A. *Lexical-Syntactical Patterns for Subjectivity Analysis of Social Issues*[M]// *Active Media Technology* Springer International Publishing, 2013:241-250.
- [29] T. Wilson, J. Wiebe, and R. Hwa. Just how mad are you? Finding strong and weak opinion clauses, *Proceedings of AAAI*, pp. 101-109, 2004.
- [30] Wilson T, Wiebe J, Hoffmann P. Recognizing contextual polarity in phrase-level sentiment analysis[J]. *International Journal of Computer Applications*, 2005, 7(5):347-354.

Event Multiple Influence Calculation and Relationship of Multiple Influence Discovery

Yunlan Xue^{#1}, Lingyu Xu^{#2}, Jie Yu^{#3}, Lei Wang^{#4}, Gaowei Zhang^{#5}, Qun Zhuge^{#6}

[#]*School of Computer Engineering and Science, Shanghai University*

¹xueyunlan@i.shu.edu.cn

²xly@shu.edu.cn

³jieyu@shu.edu.cn

⁴wanglei727@shu.edu.cn

⁵yiqigo0125@163.com

⁶franciszhuge@i.shu.edu.cn

Abstract—Due to the development of web technology, the research on Internet big data become more and more popular. However, information presents polymorphism and complexity under the status of the rapid development of Internet big data. Most of the traditional methods of event influence calculation use observable data to evaluate and ignore the decay influence of event. In fact, event influence grows from the moment of event happening to the end, meanwhile, it accompanis with a certain decay. What's more, event influence presents observable influence and decay influence on different network media. This paper proposes a new method to calculate event influence on different network media and find the relationships between observable influence and decay influence on different stages of event. Experiments show that better influence calculation will be achieved by decay algorithm based on Ebbinghaus forgetting curve and information fusion by considering interaction between observable influence and decay influence.

Keywords—Internet big data; event influence; relationship; decay curve; forgetting curve

I. INTRODUCTION

Currently news event flood which comes from different websites spreads throughout the web. With the development of the techniques of Topic Detection and Tracking (TDT) [1], some web service can gather news information from different websites and structure it into news topics. Because of a lager of news events, reader is difficult to quickly find what they want to read. So ranking news events which can provide the most valuable and influential news events is a valuable research subject.

What's more, the news event is the main source of network and social public opinion place. It is important to judge news event influence for grasping the social public opinion tendency accurately and promptly, and its influence calculation has great significance on social security and other relevant aspects. The news event influence can be understood as composite of a group people that are affected by news event, the size of geographic range, and the force on social factors.

At present, news event influence calculation has the following traditional methods. Some scholars rank news event by efficiency and transfer information of it [2], and others sort web pages by the layout of page and news transfer information [3]. In addition, with the presence of the aging theory [4] for news event life-cycle modelling, the influence

of a news event is to decay over time. The biological aging rate is not a constant ratio. In the same way, the decay value of a news event influence should not be a fixed parameter. The rate of decay is actually affected by current and past situations of a news event.

In people's social activities, information production and propagation in the form of topic in most time. Researchers find that different topics have different influence, even if influence of the same topic is also different in different groups [5, 6, 7]. Some scholars rank the events by the frequencies of event reported in time units and the number of consecutive effective time units [8], and a ranking algorithm is proposed to find the most authoritative news sources and identify the most interesting events in the different categories to which news article belongs [9]. A three-step automatic online algorithm for news issue construction is proposed, and time and quantity preference are separately used for ranking news events [10]. In their next work [11], an automatic online news topic rank algorithm is proposed based on inconsistency analysis between media focus and user attention. However, it only considers the constant decay in the aging model.

However, the rapid development of Internet big data under the new status, and information presents polymorphism and complexity, event influence presents different trajectories on different stages. The event influence may present observable influence and decay influence, how to quantify the event influence and find the relationships between observable influence and decay influence, which are complementary or alternative relationship. In this paper, we aim to reveal event multiple influence and relationships of them in Eastmoney and Baidu, the biggest financial website and the biggest search engine in China.

The rest of the paper is organized as follows. We introduce preliminary concepts in Section 2. Extensive experimental results are presented in Section 3. We conclude the paper in Section 4.

II. PRELIMINARY CONCEPTS

1) Event Influence Description

The so-called event occurs in a particular time or place by one or more roles, which is composed of one or more actions, it means an action or state change. Event is the unit of

people understand and experience of the world, and meet people's normal cognition rule.

Event presents a variety of trajectories under the rapid development of network media. We choose stock news events in the field of finance. The news events cause official media and mainstream forums search and comments in Cyberspace. News events move by respective trajectories in different media, the calculation of event influence is not the same on different stages. Event has different observable influence and decay influence on different stages in different media. Our main work is how to calculate the event influence and find the relationships between observable influence and decay influence.

2) Observable Influence of Event

With the rise of Internet media, people are used to comment about one event on network carriers, while Internet carriers record information of people's living, working and studying. The current relatively popular network media contain microblog, Twitter, Facebook, forums, etc., which have thousands of users. When a hot event happens, a lot of views and comments about the event have diffusion fast on network to form a powerful network influence. The research and analysis of network events emerge in endlessly [12, 13, 14, 15, 16, 17, 18, 19]. In particular, investors publish their views and emotions when a good or a bad event happens in financial field.

$$Inf_t^{observable} = Post_t \quad (1)$$

where $Inf_t^{observable}$ is the observable influence of event series. $Post_t$ is the main observable influence of event. It is element of the action, and mean the change process and its characteristics of events, which is the degree of movement, description of the way, method and so on.

3) Decay Influence of Event

Event influence grows from the moment of event happening, which accompany with decay at the same time. The decay curve was first introduced by Ebbinghaus [20] as formula 2. The s is the strength of memory, which controls how fast we forget about events. The R decreases memory while time elapses from the happening of event.

$$R = e^{-\frac{t}{s}} \quad (2)$$

There are also some variants of forgetting curve such as the S-shaped curve [21] and the power-law curve. Ebbinghaus proposes memory forgetting curve that is memory on a certain event decreases while time elapses from the happening of event, the velocity of the curve from fast to slow. Traditional methods of calculating information influence use different decay schemes to reduce the noise of information. Liang Kong ranks news event by influence decay and information fusion [22], and the interest-forgetting curve for music recommendation use Ebbinghaus memory forgetting curve [23].

In this paper, we use the Baidu search index curve as decay curve to estimate the decay velocity of event influence, because the Baidu search index has the same feature as traditional information, such as twitter, microblog.

We test the similarity of official search index and Ebbinghaus memory forgetting curve. The Fig.1 is the original *SearchIndex* curve after normalization. The fit curve of *SearchIndex* of 000799 in Baidu in cycle life of the Plasticizer event (from 19/11/2012 to 16/1/2013) accord with the change rule of Ebbinghaus memory forgetting curve.

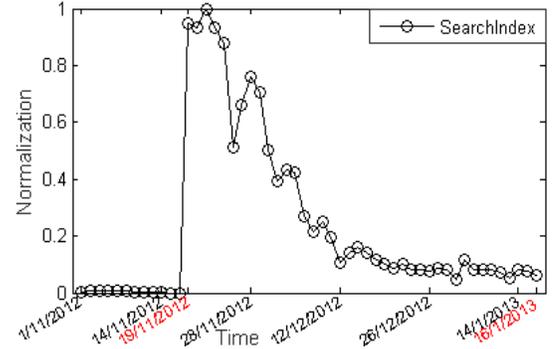


Fig1. Baidu search index

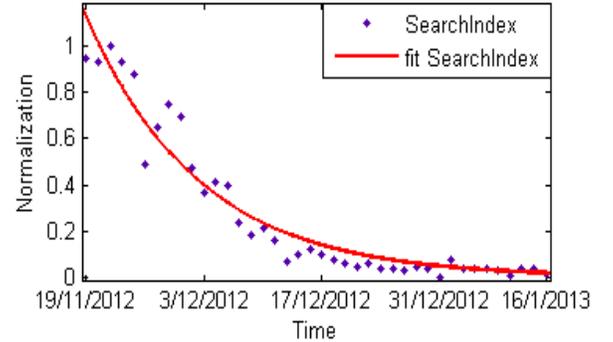


Fig2. The fit curve of Baidu search index

The fit curve of *SearchIndex* is power law curve as Fig.2, its function is $f(t) = \alpha \cdot e^{\beta \cdot t}$. $\alpha = 1.252$,

$\beta = -0.1044$. The 95% confidence bounds of α is between 1.13 and 1.373, and the 95% confidence bounds of β is between -0.118 and -0.09075. Goodness of fit are as follow: SSE is 0.262, R-square is 0.9357, Adjusted R-square is 0.934, RMSE is 0.08303.

This paper combines data of Baidu to form our decay curve to model the effects in people's comments. Therefore, we bring forward the concept of decaying curve which is expressed as formula 3.

$$Inf_t^{decay} = SearchIndex_t \quad (3)$$

Where Inf_t^{decay} is the decay influence of event. t is the life cycle from the happening of an event till ending.

SearchIndex is the search index of one stock from Baidu, and it means the decay influence of event. The formula 3 characterizes how much influence of a happened event is retained by now.

4) Patterns of Time Series

This paper gives some definitions of time series pattern due to the idea of paper [24].

DEFINITION 1. (**Uptrend Pattern**), given a time series $P = \langle p_1, p_2, \dots, p_w \rangle$ with the window threshold w , while $p_1 = \min \langle p_1, p_2, \dots, p_w \rangle$ and $p_w = \max \langle p_1, p_2, \dots, p_w \rangle$, this time series P as uptrend pattern, which is P_U .

DEFINITION 2. (**Downtrend Pattern**), given a time series $P = \langle p_1, p_2, \dots, p_w \rangle$ with the window threshold w , while $p_1 = \max \langle p_1, p_2, \dots, p_w \rangle$ and $p_w = \min \langle p_1, p_2, \dots, p_w \rangle$, this time series P as downtrend pattern, which is P_D .

5) Event Influence

We calculate event influence combine observable influence and decay influence. The observable influence and decay influence effect event interaction on the beginning stage of the life cycle when their curve patterns are different; while, only one of observable influence or decay influence effect event on the ending stage when their time series patterns are same.

$$inf_t^{event} = \begin{cases} inf_t^{observable} + inf_t^{decay}, & P(inf_t^{observable}) \neq P(inf_t^{decay}); \\ inf_t^{observable}, & P(inf_t^{observable}) = P(inf_t^{decay}). \end{cases} \quad (4)$$

where inf_t^{event} is the event influence, the complementary or alternative relationships between observable influence and decay influence under the constraint of pattern. When $P(inf_t^{observable}) \neq P(inf_t^{decay})$, the observable influence and decay influence is complementary relationship, we combine $inf_t^{observable}$ and inf_t^{decay} to calculate the event influence. And $P(inf_t^{observable}) = P(inf_t^{decay})$, the observable influence and decay influence is alternative relationship, we choose the observable influence as the event influence.

III. EXPERIMENT AND ANALYSIS

1) Detection the cycle life of event

Traditional methods to detect and track events contain the keyword of event detection [1]. C. Chen proposes an aging theory to model the life cycle of event [4]. The thought of TDT and aging theory are employed in our work. If we calculate event influence, we need to know the life cycle of event. In this paper, we define the cycle life of event that is from the begin to the end of event. The begin is from when the topic keyword appear in the context last Δt days, and the end is when the topic keyword disappear last Δt days.

This paper, we choose the Plasticizer event of Jiuguijiu in 19/11/2012. We crawl the posts of Jiuguijiu (000799) from Eastmoney website from 2012 to 2013. We extract the topic keyword of Plasticizer event by TF-IDF method, and the time

cycle of Plasticizer in the post is from 19/11/2012 to 16/1/2013 based on our definition of the cycle life of event. So the life cycle of the Plasticizer event is from 19/11/2012 to 16/1/2013 as Fig.3.

2) The event influence calculation fusion decay curve

In this paper, we crawl the post from Eastmoney website from 2012 and 2013. The observable curve fuse decay curve to calculate event influence, the observable curve uses number of post and the decay curve uses *SearchIndex*. Fig.4 is the original data about *SearchIndex* and number of *Post* after normalization.

We need to judge the patterns of observable curve and decay curve in the cycle life of the Plasticizer event by definition 1 and definition 2. We define $w = 3$ according with experience value. Fig.5 is the patterns of *SearchIndex* and *Post*. The $Pattern(SearchIndex) = P_D$ and $Pattern(Post) = P_U$ are from 19/11/2012 to 29/11/2012, while $Pattern(SearchIndex) = P_D$ and $Pattern(Post) = P_D$ are from 30/11/2012 to 16/1/2013.

We calculate event influence according to the formula 4. When the patterns of curve are different, we need to consider the acting force of decay curve of information. This time, we fuse observable influence and decay influence to calculate event influence.

The condition of $P(inf_t^{observable}) \neq P(inf_t^{decay})$ is from 19/11/2012 to 29/11/2012, the Plasticizer event effect *SearchIndex* and *Post*, and the event influence is equal to the sum of $inf_t^{observable}$ and inf_t^{decay} . The condition of $P(inf_t^{observable}) = P(inf_t^{decay})$ is from 30/11/2012 to 16/1/2013, and the event influence can use $inf_t^{observable}$ or inf_t^{decay} , because of $inf_t^{observable}$ and inf_t^{decay} are alternative. As is shown in Fig.6.

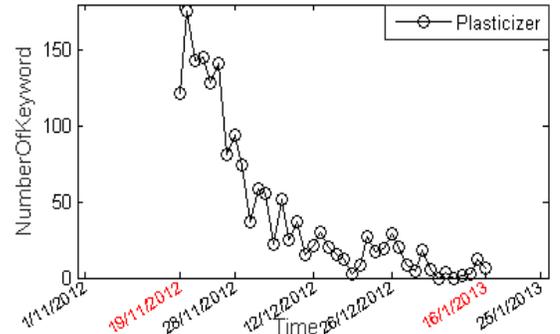


Fig3. Keyword of Plasticizer is from 19/11/2012 to 16/1/2013

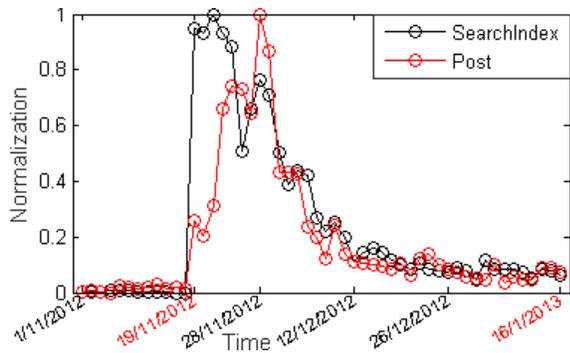


Fig4. The time series of *SearchIndex* and *Post*

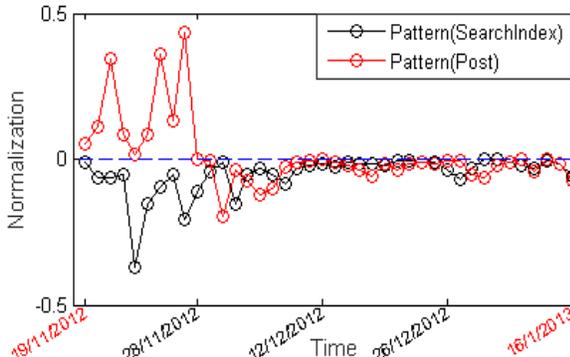


Fig5. The *Pattern(SearchIndex)* and *Pattern(Post)*

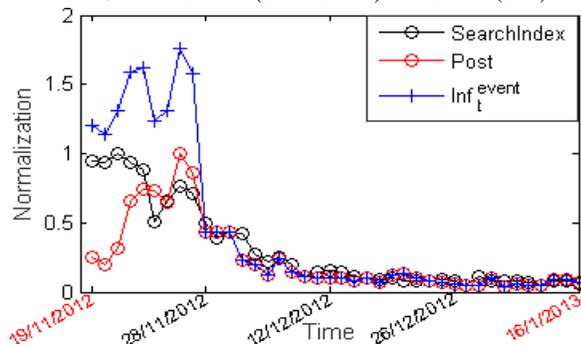


Fig6. The time series of event influence

IV. CONCLUSIONS

In this paper, we calculated event influence in a new method and found the relationships between observable influence and decay influence on different stages in the life cycle of event. We not only took both observable influence and decay influence into consideration, but also brought in the interaction between them to our framework. Experimental results showed that our method as a typical scheme using both observable influence and decay influence impact, and found that observable influence and decay influence work together on event in the beginning time and they are alternative relationship in the ending of event. At the same time, their cut-off point is very apparent.

As a future work, we will consider multiple spaces feature of event in the condition of Internet big data to improve our method, since such features have been widely used in hot news event.

REFERENCES

- [1] J. Allan, J. Carbonell, G. Doddington, J. Yamron, and Y. Yang. Topic Detection and Tracking Pilot Study: Final Report. In Proceedings of the DARPA Broadcast News Transcription and Understanding Workshop, volume 1998, pages 194–218, 1998.
- [2] Del Corso GM, Gullí A, Romani F. Ranking a stream of news. In: Ellis A, Hagino T, eds. Proc. of the WWW 2005. New York: ACM, 2005. 97–106.
- [3] Yao JY, Wang J, Li ZW, Li MJ, Ma WY. Ranking Web news via homepage visual layout and cross-site voting. In: Lalmas M, ed. Proc. of the ECIR 2006. Heidelberg: Springer-Verlag, 2006. 131–142.
- [4] C. Chen, Y. Chen, Y. Sun, and M. Chen. Life Cycle Modeling of News Events using Aging Theory. In ECML'03, pages 47–59, 2003.
- [5] Yunlan Xue, Lingyu Xu, Jie Yu, Lei Wang, Gaowei Zhang, An Estimate Method of Event Influence Scope based on Special Field, 2015 International Conference on Identification, Information & Knowledge in the Internet of Things, Pages: 39 - 44
- [6] Gruhl D, Guha R, Liben-Nowell D, Tomkins A. Information diffusion through blogspace // Proceedings of the 13th International Conference on World Wide Web. New York, USA, 2004: 491-501.
- [7] Degan Zhang, Guang Li, Ke Zheng. An energy-balanced routing method based on forward-aware factor for Wireless Sensor Network. IEEE Transactions on Industrial Informatics, 2014,10(1):766-773
- [8] Yunlan Xue, Lingyu Xu, Jie Yu, Lei Wang, Gaowei Zhang, A Detection Method of Stock Event Source, The 11th International Conference on Semantics, Knowledge and Grids on Big Data, Pages: 174 - 179
- [9] Zheng Xu et al. Knowle: a Semantic Link Network based System for Organizing Large Scale Online News Events. Future Generation Computer Systems, 2015, 43-44, 40-50.
- [10] C. Wang, M. Zhang, S. Ma, and L. Ru. Automatic Online News Issue Construction in Web Environment. In WWW'08, pages 457–466, 2008.
- [11] C. Wang, M. Zhang, L. Ru, and S. Ma. Automatic Online News Topic Ranking using Media Focus and User Attention based on Aging Theory. In CIKM'08, pages 1033–1042, 2008.
- [12] Wang Lei, Yu Jie, Xu Lingyu, Xue Yunlan, Zhang Gaowei, Consistency research between virtual space and real space based on an event-driven condition, Source: Journal of Computational Information Systems, v 11, n 7, p 2345-2354, April 1, 2015
- [13] C. C. Aggarwal and K. Subbian. Event detection in social streams. In SDM'12, pages 624–635, 2012.
- [14] H. Becker, M. Naaman and L. Gravano. Beyond trending topics: Real-world event identification on twitter. In ICWSM'11, pages 438–441, 2011.
- [15] R. Li, K. H. Lei, R. Khadiwala, and K. C.-C. Chang. Tetas: A twitter-based event detection and analysis system. Data Engineering, International Conference on, 0:1273–1276, 2012.
- [16] S. Petrovic, M. Osborne, and V. Lavrenko. Streaming first story detection with application to twitter. In HLT'10, pages 181–189, 2010.
- [17] J. Weng and B.-S. Lee. Event detection in twitter. In ICWSM'11, 2011.
- [18] Q. Zhao, P. Mitra and B. Chen. Temporal and information flow based event detection from social text streams. In AAAI'07, pages 1501–1506, 2007.
- [19] Xiangmin Zhou, Lei Chen, Event detection over twitter social media streams, The VLDB Journal — The International Journal on Very Large Data Bases, Volume 23 Issue 3, June 2014.
- [20] H. Ebbinghaus. Memory; A Contribution to Experimental Psychology. New York by Teachers College, Columbia University, 1885.
- [21] L. Averell and A. Heathcote. The form of the forgetting curve and the fate of memories. Journal of Mathematical Psychology, 55:25–35, 2011.
- [22] Liang Kong, Shan Jiang, Rui Yan, Shize Xu, Yan Zhang, Ranking News Events by Influence Decay and Information Fusion for Media and Users, CIKM'12, October 29–November 2, 2012, Maui, HI, USA. Copyright 2012 ACM 978-1-4503-1156-4/12/10.
- [23] Jun Chen Chaokun Wang Jianmin Wang, Modeling the Interest-Forgetting Curve for Music Recommendation, MM'14, November 03 - 07 2014, Orlando, FL, USA. Copyright 2014 ACM 978-1-4503-3063-3/14/11.
- [24] Fu-Lai Chung, Tak-Chung Fu, Vincent Ng, and Robert W. P. Luk, An Evolutionary Approach to Pattern-Based Time Series Segmentation, IEEE transactions on evolutionary computation, vol. 8, October 2004.

Secure Outsourcing Algorithm of Polynomials in Cloud Computing

Xianlin Zhou
Collage of Mathematics
and Software Science,
Sichuan Normal University,
Chengdu, Sichuan, P. R. China
flyinfo@qq.com

Zheng Xu *
The Third Research Institute of
the Ministry of Public Security,
Shanghai, China
zhengxu@shu.edu.cn

Yong Ding, Zhao Wang, Xiumin Li
School of Computer Science and Information Security,
Guilin University of Electronic Technology,
Guilin, Guangxi, P. R. China
stone_dingy@126.com, 824420896@qq.com,
630717447@qq.com

Jun Ye *
Sichuan University of Science & Engineering,
Artificial Intelligence Key
Laboratory of Sichuan Province,
Zigong, Sichuan, P. R. China
yeyun@suse.edu.cn

Abstract— In the era of information explosion, people have to deal with huge amount of data. It is a great computation burden for the resource-constrained clients. Cloud computing connects large amounts of network resources, and forms a vast pool of resources. It provides much convenience for people. Clients can outsource the complex computation task to the powerful cloud server. In this way, the computation burden of clients can be greatly reduced. In this paper a new algorithm of secure outsourcing for polynomials is proposed. In the computation process, the computation polynomial is hidden to cloud server, and the inputs and outputs of polynomials will not revealed. In addition, clients can verify the result easily.

Keywords— Secure Outsourcing; Cloud Computing; Polynomial

1 Introduction

With the rapid development of society, we have entered an era of information explosion. People have to face the huge amount of data. It brings much trouble to handle the complex computation task with huge data.

Cloud computing [19, 10, 22, 17, 5] shares the computing resources, which provides a lot of convenience for the resource-constrained clients. Clients can outsource the heavy computation work to cloud server with pay-per-use manner. And the

computation cost of clients can be greatly reduced. Especially in big data era, cloud computing is a perfect technique, which helps people deal with big data [15, 12, 14]. The high reliability, strong processing capacity, large storage space of cloud computing make the clients with limited storage capacity and computing power to remotely operate the big data [17, 18]. The framework of big data based on cloud computing, realize a distributed operating system by utilizing the powerful computation and storage. It provides the efficient data access [16, 21]. Although there are many advantages, the development and application of cloud computing are seriously constrained by the data security and privacy issues. In cloud computing, the resource-constrained clients conveniently obtain the results from the cloud servers. However, it is hard to find a reliable cloud service provider in cloud computing. The cloud server may want to obtain the inputs and outputs of clients. In order to get more benefits and save the resources, cloud server will probably return a random result without computing. And the wrong results would be returned due to the loophole of software and the fault of hardware. Although some solutions in this aspect are studied, the verification is unsatisfactory. This forms the notion of verifiable computation [7, 20, 13, 4, 9], which enables clients can check the correctness of the returned results.

In 2002, secure outsourcing for scientific computing is studied by Atallah et al. [1]. They pro-

posed some camouflage technologies to protect the privacy of the outsourced computations. However, these does not solve the problem of the verifiability for computing results. In 2005, Hohenberger and Lysyanskaya [11] proposed the formal security definition of outsourcing. In 2008, Benjamin and Atallah [3] used homomorphic encryption to construct a secure outsourcing scheme for linear algebraic computation, in which the client can verify the results. And in 2009, based on ideal lattices, a fully homomorphic encryption scheme was proposed by Gentry [8]. However, the efficiency is low.

Polynomials are often used in many application fields, such as, signal processing, data analysis etc. In 2011, Benabbas et al. [2] proposed an algorithm of secure outsourcing for polynomials based on homomorphic encryption. In 2012, Fiore and Genaro [6] proposed a scheme for verifiable delegation of large polynomials. However, in these two schemes, the inputs would be revealed. In 2016, Ye et al. [20] proposed a scheme for secure outsourcing polynomials, in which an extra polynomial will be outsourced for verification.

Our Contributions The main contributions are as follows, the transformation technique and the secure scheme for secure outsourcing of polynomials. The transformation technique guarantees the security of the outsourcing polynomial, the cloud server cannot get the real polynomial which will be computed. In the secure outsourcing scheme, the inputs and outputs are keeping privacy, and client can easily verify the returned results.

1.1 Organization of this Paper

The organization of this paper is as follows. Some preliminaries are given in Section 2. The algorithm of secure outsourcing for polynomials is given in Section 3. Then in Section 4 we give the security analysis. Finally, the conclusion is made in Section 5.

2 Preliminaries

Outsource-security: An algorithm is said to be an outsource-secure algorithm if:

Correctness. The result returned from the cloud

server is the correct implementation of the algorithm.

Security. For all probabilistic polynomial-time adversary, the original computation cannot be obtained from the outsourced disguised computation.

Verifiable Outsourcing Computation: We follow the definition in [20].

A verifiable outsourcing computation scheme is defined by the following algorithms:

KeyGen(f, k) \rightarrow (PK, SK): Based on the security parameter k , the key generation algorithm generates a key pair (PK, SK) for the function f . PK is provided to the server, and client keeps SK .

ProGen(x) \rightarrow (σ_x, V_x): The problem generation algorithm is run by client, who uses SK to encode the input x as σ_x which is given to server, and a verification key V_x which is kept private by client.

Compute(σ_x) \rightarrow (σ_y): The algorithm is run by the server to compute an encoded version of the output σ_y .

Verify(V_x, σ_y) \rightarrow ($y \cup \perp$): The algorithm returns the value $y = f(x)$ or \perp indicating that σ_y does not equal to $f(x)$.

A verifiable computation scheme should be correct, secure and efficient.

3 Secure Outsourcing of Polynomials

We consider the following scenario. A resource-constrained client wants to outsource a high degree polynomial with fixed coefficients. This polynomial will be used latter for some applications frequently. The polynomial, the inputs and the outputs should be blind to cloud server. And the client should verify the correctness of the result efficiently.

3.1 Design Goals

To securely outsourcing the computation of polynomials efficiently, the design goals of our system are as follows.

- **Disguise.** To design a transformation method which disguise the real computing polynomials, so that the cloud server cannot get the information of original polynomials.

- **Privacy Preserving.** To prevent the cloud server from learning the information of the inputs and outputs.
- **Verification.** To make sure that the server returns the correct computing results.
- **Efficiency.** The computation cost of verification should be greatly less than that of polynomial computation.

3.2 System Model

There are two parties in the model, a resource-constrained client and a untrusted powerful cloud server. The system model is shown in Fig. 1.

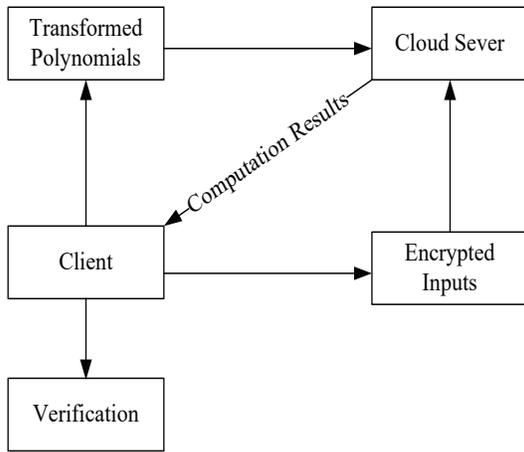


Figure 1: System Model

Client firstly transform the original polynomial into a disguised polynomial, and outsources the disguised polynomial to the cloud server. When client wants to do some computation on the polynomial, he/she outsources the encrypted inputs to the cloud servers. After the computing, cloud server returns the computation results to the client. Then client verify the computation results. If the returned results are correct, client will transform the returned results into the real computation results.

In the following we give the polynomial transformation technique and generate our outsourcing method based on the technique in [2].

3.3 Transformation Technique

The polynomial

$$f(x) = a_0 + a_1x + a_2x^2 + \cdots + a_nx^n$$

where $a_i \in \mathbb{Z}_p, 0 \leq i \leq n$ is a high degree polynomial, will be outsourced to cloud server. Client want to compute the function on the value of x .

For the secure outsourcing and efficient verification, a transformed polynomial $F(\sigma_x)$ is constructed.

Client selects $r \in_R \mathbb{Z}_p, c \in_R \mathbb{Z}_p$ and $d \in_R \mathbb{Z}_p$, and computes

$$b_0 = c + a_0.$$

The coefficients of transformed polynomial

$$F(\sigma_x) = b_0 + b_1\sigma_x + b_2\sigma_x^2 + \cdots + b_n\sigma_x^n$$

is generated as

$$b_i = a_i r^i - d^i$$

where $1 \leq i \leq n$. $F(\sigma_x)$ will be outsourced to cloud server.

3.4 Our Scheme

We assume σ_x is an encoded input and σ_y is an encoded output, the polynomial F is a encrypted function.

Client wants to computes

$$f(x) = a_0 + a_1x + a_2x^2 + \cdots + a_nx^n \pmod{p}$$

where $f(x)$ is a high degree polynomial. He/she firstly transforms $f(x)$ into $F(\sigma_x)$ by using the transform technique, and then delegates it to the cloud server. And the client can verify the correctness of the result.

Initialization. Client randomly selects six numbers r, c, d, R, k_0 and k_1 , then client generates

$$F(x) = b_0 + b_1x + b_2x^2 + \cdots + b_nx^n$$

where

$$b_0 = a_0 + c$$

and

$$b_i = a_i r^i - d^i$$

$i = 1, 2, \dots, n$.

Then client computes

$$t_0 = g^{k_0 + Rb_0}$$

$$t_1 = g^{k_0 + k_1^i + Rb_i}$$

where $i = 1, 2, \dots, n$.

Delegation. We denote $t = (t_0, t_1, \dots, t_n)$. Client sends the polynomial

$$F(x) = b_0 + b_1x + b_2x^2 + \dots + b_nx^n$$

and

$$t = (t_0, t_1, \dots, t_n)$$

to cloud server.

When client wants to compute $f(x)$, client computes $\sigma_x = \frac{x}{r}$ and sends σ_x to cloud server. Then client computes

$$Z = \prod_{i=0}^n t_1^{\sigma_x^i} = g^{k_0 \frac{1 - (k_1 \sigma_x)^{n+1}}{1 - k_1 \sigma_x}}.$$

Computation. Cloud server computes

$$\sigma_y = F(\sigma_x)$$

and

$$T = \prod_{i=0}^n t_1^{\sigma_x^i}.$$

Then cloud server sends (σ_y, T) to client.

Verification. Client verifies whether following equation holds

$$T = Zg^{R\sigma_y}.$$

If not, the server gives the wrong answer, σ_y is not correct. If the equation holds, client can get the final result by computing

$$y = \sigma_y - y_1 \pmod{p}$$

where

$$\begin{aligned} y_1 &= \sum_{i=1}^n d^i \sigma_x^i - c \\ &= \frac{d\sigma_x - (d\sigma_x)^{n+1}}{1 - d\sigma_x} - c. \end{aligned}$$

4 Security Analysis

Theorem 1. *The input and output of the polynomial are secure.*

Proof. The real input is x , however, the encrypted input is σ_x , where $\sigma_x = \frac{x}{r}$. For r is randomly chosen, the input x is keeping privacy.

The output client needs is $y = \sigma_y - y_1$. Cloud server can get σ_y , however, it cannot get y_1 .

For

$$y_1 = \frac{d\sigma_x - (d\sigma_x)^{n+1}}{1 - d\sigma_x} - c$$

where d and c are randomly chosen by client. Cloud server can just get σ_x . And in the computation process, cloud server cannot get the private parameters d and c from the outsourced polynomial.

Hence, the input and output would not revealed. \square

5 Conclusion

In big data era, people cannot afford the more and more complex computation work due to the constrained computation resources. Outsourcing computation helps people to solve the heavy computation task. In this paper, a new algorithm for secure outsourcing of high degree polynomials is given. And we introduce the transformation technique, with which the real polynomial will be hidden to the untrusted cloud server. In addition, the input and output will not be revealed in the computation process. Finally, the clients can easily verify the returned result.

ACKNOWLEDGMENT

This work was supported in part by the Science Founding of Artificial Intelligence Key Laboratory of Sichuan Province (2014RYJ06), in part by the Scientific Research Fund Project of Sichuan Normal University (15YB008), in part by the Guangxi experiment center of information science Foundation, in part by the Innovation Project of Guangxi Graduate Education(XJYC2012020), in part by the National Science and Technology Major Project

under Grant (2013ZX01033002-003), in part by the National High Technology Research and Development Program of China (863 Program) under Grant (2013AA014601), in part by the National Science Foundation of China under Grant (61300202), and in part by the Science Foundation of Shanghai under Grant (13ZR1452900).

References

- [1] M.J. Atallah, K.N. Pantazopoulos, J.R. Rice, and E.E. Spafford. Secure outsourcing of scientific computations. *Advances in Computers*, 54:215–272, 2002.
- [2] S. Benabbas, R. Gennaro, and Y. Vahlis. Verifiable delegation of computation over large datasets. In *Advances in Cryptology—CRYPTO 2011*, pages 111–131. Springer, 2011.
- [3] D. Benjamin and M.J. Atallah. Private and cheating-free outsourcing of algebraic computations. In *Sixth Annual Conference on Privacy, Security and Trust, PST 2008, Fredericton, New Brunswick, Canada*, pages 240–245. Springer-Verlag, October 2008.
- [4] S.G. Choi, J. Katz, R. Kumaresan, and C. Cid. Multi-client non-interactive verifiable computation. In *Proc. of the 10th Theory of Cryptography Conference, TCC 2013, Tokyo, Japan*, volume 7785, pages 499–518. Springer-Verlag, March 2013.
- [5] S. El-Sayed, H. Kader, M. Hadhoud, and D. Abdelminaam. Mobile cloud computing framework for elastic partitioned/modularized applications mobility. *International Journal of Electronics and Information Engineering*, 1(2):53–63, 2014.
- [6] D. Fiore and R. Gennaro. Publicly verifiable delegation of large polynomials and matrix computations, with applications. In *Proc. of the 2012 ACM conference on Computer and communications security, ACM New York, NY, USA*, pages 501–512. Springer-Verlag, October 2012.
- [7] R. Gennaro, C. Gentry, and B. Parno. Non-interactive verifiable computing: Outsourcing computation to untrusted workers. In *Advances in Cryptology—CRYPTO 2010*, volume 6223, pages 465–482. Springer, 2010.
- [8] Craig Gentry et al. Fully homomorphic encryption using ideal lattices. In *STOC*, volume 9, pages 169–178, 2009.
- [9] S.D. Gordon, J. Katz, F. Liu, E. Shi, and H. Zhou. Multi-client verifiable computation with stronger security guarantees. In *Theory of Cryptography*, pages 144–168. Springer, 2015.
- [10] I.A.T. Hashem, I. Yaqoob, N.B. Anuar, S. Mokhtar, A. Gani, and S.U. Khan. The rise of “big data” on cloud computing: Review and open research issues. *Information Systems*, 47:98–115, 2015.
- [11] S. Hohenberger and A. Lysyanskaya. How to securely outsource cryptographic computations. In *Proc. of the second Theory of Cryptography Conference, TCC 2005, Cambridge, MA, USA*, volume 3378, pages 264–282. Springer-Verlag, February 2005.
- [12] C Hu, Z. Xu, Y. Liu, L. Mei, L. Chen, and X. Luo. Semantic link network-based model for organizing multimedia big data. *IEEE Transactions on Emerging Topics in Computing*, 2(3):376–387, 2014.
- [13] B. Parno, J. Howell, C. Gentry, and M. Raykova. Pinocchio: Nearly practical verifiable computation. In *Proc. of the 2013 IEEE Symposium on Security and Privacy, IEEE Computer Society Washington, DC, USA*, pages 238–252. Springer-Verlag, March 2013.
- [14] Z. Xu, Y. Liu, L. Mei, C. Hu, and L. Chen. Semantic based representing and organizing surveillance big data using video structural description technology. *Journal of Systems and Software*, 102:217–225, 2015.
- [15] Z. Xu, Y. Liu, N. Yen, L. Mei, X. Luo, X. Wei, and C. Hu. Crowdsourcing based description of urban emergency events using social media big data. *IEEE Transactions on Cloud Computing*, 2016. DOI: 10.1109/TCC.2016.2517638.
- [16] C.g Yang and J. Ye. Secure and efficient fine-grained data access control scheme in cloud computing. *Journal of High Speed Networks*, 21(4):259–271, 2015.
- [17] J. Ye, X. Chen, and J. Ma. An improved algorithm for secure outsourcing of modular exponentiations. In *Proc. of 29th International Conference on Advanced Information Networking and Applications Workshops (WAINA), 24-27, March*, pages 73–76. IEEE, 2015.
- [18] J. Ye, M. Miao, P. Chen, and X. Chen. Request-based comparable encryption scheme with multiple users. In *10th International Conference on Broadband and Wireless Computing, Communication and Applications, BWCCA 2015, Krakow, Poland, November 4-6, 2015*, pages

- 414–416, 2015.
- [19] J. Ye and J. Wang. Secure outsourcing of modular exponentiation with single untrusted server. In *Proc. of 18th International Conference on Network-Based Information Systems (NBIS), 2-4, September*, pages 643–645. IEEE, 2015.
- [20] J. Ye, H. Zhang, and C. Fu. Verifiable delegation of polynomials. *International Journal of Network Security*, 18(2):283–290, 2016.
- [21] J. Ye, W. Zhang, S. Wu, Y. Gao, and J. Qiu. Attribute-based fine-grained access control with user revocation. In *Information and Communication Technology*, pages 586–595. Springer, 2014.
- [22] D. Zissis and D. Lakkas. Addressing cloud computing security issues. *Future Generation Computer Systems*, 28(3):583–592, 2012.

Deep Data Analyzing Application Based on Scale Space Theory in Big Data Environment

Ye Hu

College of Mechanical Engineering
Ningbo Dahongying University
Ningbo, China
Tongji University, Shanghai, China
pxhuve@126.com

Kun Gao

Zhejiang Business Technology
Institute, China
Zhejiang Wanli University
Ningbo, China
kungao@live.com

Zheng Xu

Tsinghua University, Beijing, China
The Third Research Institute of the
Ministry of Public Security,
Shanghai, China
xuzheng@shu.edu.cn

Abstract—This paper introduces the basic scientific idea of the multi-scale to the field of big data analyzes, proposes a multi-scale framework of data analyzes in big data environment, present the multi-scale algorithm framework of knowledge conversion theory and apply the algorithm framework to the multi dimension association rules analysis. The proposed multi-scale association rule analysis algorithm uses the benchmark data set of analyzing results and the influence weight of benchmark data sets for target scale data sets to derived the association rules behind object scale data set, realize knowledge across scales derived and provide the possibility for multi-scale decision.

Keywords—Scale Space; Big Data; Frequent Item-set; Association Rules;

I. INTRODUCTION

Multi scale is a common phenomenon in the objective world, in recent years has been widespread concern in academic circles, and gradually developed into an independent research topic - multi-scale science. Scholars of mathematics, physics, chemistry and other fields have been introduced the multi-scale theory into their own discipline and carry out a series of related research [1]. The rapid development of data fusion technology [2] and the associated model [3] application has greatly promoted the research of multi-scale domains; the collection, transmission, synthesis, filtering, correlation and synthesis of multiple information sources greatly reduce the time consumption of the scale conversion, and improve the accuracy of the data results.

Aiming at the multi-scale research on the field of data analysis, [4] using inherent multi-scale characteristic and concept hierarchy of spatial data, proposed point-object association rules analysis algorithm oriented multi-scale spatial, realized the scaling up of association rules in spatial data. [5] In conjunction with scale conversion mechanism of geography science field, proposed multi-scale clustering analysis algorithm based on enhanced weighted vector. [6] uses hierarchical multi-scale combination classification method, proposed an adaptive multi-scale segmentation combined classification algorithms, reduces the training time and get a better performance of the classifier. For the problem of the unstable of long range dependence in the existing self-similar network, [7] built network flow model and flow prediction based on discrete wavelet transform multi-scale analysis. Based on Least mean fast Wavelet Transform, considering the characteristics of the input flow and redundant wavelet transform, [8] carry out multi-

scale analyzing and forecasting for self-similar network traffic. Derived from the improved Mahalanobis distance measurement to conduct fast multi-scale clustering, [9] realized data segmentation model of electronic back scatter diffraction through the presentation of data quaternion. [10] Generate an image map and track information table by scanning multi-scale data, and conduct analysis for frequent pattern and multi-scale events in the type of Depth-First-Search. In [11], an iterative and interactive hierarchical multi-scale classifier is proposed to realize the multi-scale segmentation of remote sensing images, and the classification accuracy is improved compared with the general segmentation algorithm; Based on conditional random field, the context classification of multi temporal and spatial scales of geo optical remote sensing images with different periods and resolutions was realized.

With the arrival of the era of big data, the research on the application of data analyzing is more and more in-depth, and basically committed to exploring new technologies, new methods in efficiency and accuracy to achieve across. The research of multi-scale data analysis has just started, and it is very important and urgent to explore the multi-scale data analysis method which can deal with massive data, so as to support multi-scale decision making and improve the efficiency and accuracy.

II. THEORETICAL FOUNDATION

The task of Multi-scale data analyzing is to use scaling mechanism to anti-conduct the analysis results from benchmark scale data set to other target scale data set. This section will detail describe the scaling mechanism, namely the theoretical foundation of knowledge scaling [16-19].

Scaling method most commonly used method is based on spatial statistics method, to estimate the statistical law is an optimization technique, the basic assumption is built on the spatial correlation of the prior model[19, 20]. The main idea of the relevant parameters related parameters sampling points are closer than distant sampling points are more similar, and the same degree of similarity or the size of its random spatial covariance, you can be certain lag by comparing the distance separated different values of the variables and multiple scales of measurement variability regionalized random variables to determine, and therefore become a good solution for scale conversion[21-23].

A. The general essence of multi scale data analysis

Kriging interpolation method is based on the structural information sampling data reflect regional variables (variogram and covariance functions provided), based on limited data sampling point to be estimated in the neighborhood of the point or block segment, consider sample points positional relationship between space, estimated to be the point of spatial relationships and treat an estimated optimal point unbiased estimate, and gives the estimation accuracy[24-26]. Because of different purposes and conditions, have produced a variety of Kriging method: to meet the steady (or intrinsic) the assumption of second order, the use of ordinary Kriging method; in non-stationary phenomenon, the available pan-g Rigby law; in the calculation of recoverable reserves, to use non-linear estimator, you can use disjunctive kriging; when regionalized variables obey a logarithmic normal distribution, the number of available kriging; when comparing data little, small distribution rules, but it does not require the estimation accuracy of available random Kriging method[27].

Kriging is the essence of valuation to be estimated by weighted summation point near point, so that the core of the weighting coefficient λ is determined. Spatial data refers to data indicating the location of spatial entities, shape, size and distribution of many aspects of information, it has a characteristic time scale, spatial scale relations. Mutual positional relationship between the data space, and have some distribution[28]. Therefore, the use of Kriging spatial scale data push, push down is very appropriate[29].

For general data set, in theory it does not have significant spatial and time scales characteristic[30]. Because most used for classification, clustering, association rules or data set size does not attribute value of spatial data, location information of other data analysis methods. In the traditional sense of space or time scale refers to units in the study of an object or phenomenon employed. However, with further research, we found that the original definition of the scale is not accurate enough or not comprehensive enough. This paper proposes the use of the concept of hierarchical distinction between the size of the scale, in fact, it extends the concept of scale, namely the concept of layering concept has partial order can be considered to be included in scale, such as: the composition of the administrative structure of the school (school, college, professional, department, class) and the like. According to the concept of hierarchical knowledge, we can assume that any data set is a multi-scale data set of mathematical partial order. One of the most exceptional cases are juxtaposed relationship between the concept of hierarchical scale that contains the relationship between the presence of any scale are not. Strictly speaking, multi-scale of some of the data does not have practical significance, that is, in practical applications, multi-scale data are generally not of practical significance and practical significance. Through the above analysis, we can draw the essence of the scale is the size of a unit of measurement range covered. Although the scale of the performance of spatial data in time and space in the areas, in general, the performance data for other category scale, but their principles and nature has not changed. That objective scale data is weighted summation of benchmarking scale data. Therefore, after making the appropriate data set general concept hierarchy to form a multi-scale data sets generated between the sample data structure information regionalized variables, according to

the distribution of data, select the appropriate kriging, thus achieving the general scaling data.

B. The typical scaling up and scaling down method

The most typical scaling up and scaling down are respectively region ordinary Kriging method and point ordinary Kriging method, we are collectively referred to as the Kriging method. Kriging method is spatial local estimation method established on the basis of variation function theory and structural analysis, and an unbiased optimal estimation for regionalization variable aggregation in limited area. This method first defines a linear estimator:

$$P_o^*(x) = \sum_{j=1}^{m+1} \lambda_j Y(x_j) \quad (1)$$

Where, $Y(x_j)$ is the data of example point, $P_o^*(x)$ is to be estimated, λ_j is the weight for every example point, and $\sum_{j=1}^{m+1} \lambda_j = 1$. For any estimation, there exist a deviation between real value and the estimated value. $P_o^*(x)$ is a linear unbiased estimates of the optimal to the actually true value $P_o(x)$. Formula (1) is called as Kriging equation. Kriging coefficient λ can be expressed in the form of the following matrix multiplication:

$$\lambda = R^{-1}E \quad (2)$$

Following respectively introduce some point ordinary Kriging method and region ordinary Kriging method, and application mode in scaling up method and scaling down method. These foundations are the theoretical basis of the multi-scale data analysis.

1) Theoretical basis of scaling down analysis algorithm: point ordinary Kriging method

Using point ordinary Kriging method to implement scaling down, it is a key to determine the Kriging coefficient. E_{ij} is the covariance between the example point c and example point d of meta scale S in the K matrix. $f(x_i, x)$ is the covariance between sample point i in the meta scale S and target scale S' to be estimated. The detail form of E and F is as follows:

$$E = \begin{bmatrix} e_{11} & e_{12} & \dots & e_{1n} & 1 \\ e_{21} & e_{22} & \dots & e_{2n} & 1 \\ \dots & \dots & \dots & \dots & \dots \\ e_{n1} & e_{n2} & \dots & e_{nn} & 1 \\ 1 & 1 & \dots & 1 & 0 \end{bmatrix} \quad (3)$$

$$F = \begin{bmatrix} e(x_1, x) \\ e(x_2, x) \\ \dots \\ e(x_n, x) \\ 1 \end{bmatrix} \quad (4)$$

Then, the specific calculation formula of weight matrix in the scaling down analysis algorithms is as follows:

$$\lambda = \begin{bmatrix} \lambda_1 \\ \lambda_2 \\ \dots \\ \lambda_n \\ -y \end{bmatrix} = \begin{bmatrix} e_{11} & e_{12} & \dots & e_{1n} & 1 \\ e_{21} & e_{22} & \dots & e_{2n} & 1 \\ \dots & \dots & \dots & \dots & \dots \\ e_{n1} & e_{n2} & \dots & e_{nn} & 1 \\ 1 & 1 & \dots & 1 & 0 \end{bmatrix}^{-1} * \begin{bmatrix} e(x_1, x) \\ e(x_2, x) \\ \dots \\ e(x_n, x) \\ 1 \end{bmatrix} \quad (5)$$

2) Theoretical basis of scaling up analysis algorithm: domain ordinary Kriging method

Using domain ordinary Kriging method to realize scaling up, also need to determine Kriging coefficient. In this way, determine the matrix E is the same as the scaling up method in the point ordinary Kriging, that is to determine E by the covariance between each sample point from S' in the meta scale; F is determined in a completely different way, but by the impact factor to be estimated for target scale in each example point in the meta scale S'. We call this impact factor as the Influence Information, as shown in formula (6):

$$F = \begin{bmatrix} information_{10} \\ information_{20} \\ \dots \\ information_{n0} \\ 1 \end{bmatrix} \quad (6)$$

The formula for determine Kriging coefficient by using the method of domain kriging method is shown as in formula (7):

$$\lambda = \begin{bmatrix} \lambda_1 \\ \lambda_2 \\ \dots \\ \lambda_n \\ -y \end{bmatrix} \begin{bmatrix} e_{11} & e_{12} & \dots & e_{1n} & 1 \\ e_{21} & e_{22} & \dots & e_{2n} & 1 \\ \dots & \dots & \dots & \dots & \dots \\ e_{n1} & e_{n2} & \dots & e_{nn} & 1 \\ 1 & 1 & \dots & 1 & 0 \end{bmatrix}^{-1} \begin{bmatrix} information_{10} \\ information_{20} \\ \dots \\ information_{n0} \\ 1 \end{bmatrix} \quad (7)$$

Multi scale association rule analysis

We take domain Kriging method and point Kriging method as theoretical foundation of scaling up and scaling down, apply multi-scale data analysis algorithm framework in association rules analysis, and propose Association Rules Analysis Multi scale algorithm.

Multi scale data analysis algorithm framework can be summarized as three key steps:

1. Determine the benchmark BenchScale, process benchmark data sets;
2. Compared to the target scale TargetScale and benchmark Bench Scale, determined multi-scale analysis task of analysis direction, namely is to judge scaling up analysis or scaling down analysis;
3. Knowledge in the corresponding direction so the multi-scale transformation derived target scale data set behind the knowledge. The core of the algorithm is the knowledge of multi scale conversion, above has introduced in detail the scaling mechanism based on the theory, the key is to determine the Kriging equation in the weight coefficient matrix lambda, that is to say, whatever scale or scales push, we need to first determine the associated covariance matrix K benchmark data sets, and for D matrix to determine the calculation method in D for benchmark data set of object scale data set of influence factors matrix; calculated method D is a benchmark data set and target scale data set between the covariance matrix.

Multi scale data analysis algorithm framework is based on multi-scale association rule analysis algorithm. First clear association rules analysis in the target knowledge for association rules and analysis association rules in the process of time and computational resources consumed mainly in frequent item set analysis, so multi-scale association rules analysis algorithm to solve the core problem is the benchmark data set of frequent item sets derived object scale data set of frequent item sets and frequent item sets multi-scale conversion, and wrong target scale data set analysis.

So, the basic idea of multi-scale algorithm for analysis association rules is: first, clear benchmark scale BenchmarkScale, suitable frequent item set analysis algorithm for analysis benchmark data sets DataSet_{BenchmarkScale} here can choose any efficient frequent item set analysis algorithm; then clear target scale Scale_{target} and the scale of target data set dataset_{scale_{target}}, determine the scale of association rule analysis direction; finally, the calculation of the benchmark data set analysis dataset_{BenchmarkScale} results of object scale data set dataset_{Scale} knowledge of weight coefficient matrix lambda, where knowledge is frequent item sets, lambda for estimating target scale data from frequent item sets of key parameters, support. Here, we calculate the weight coefficient of standard Kriging method do some improvement: K will be designated as a benchmark data set between the similarity matrix instead of the covariance matrix, can more accurately the reaction benchmark data set between similar and correlation; push process, the D matrix specified as benchmark data set of object scale data set of impact factor matrix, in pushdown processes, matrix D specified on the grounds of domain knowledge to determine the benchmark data sets with target scale data set between the similarity matrix. The problem description and basic steps of multi scale association rule analysis algorithm are as follows:

Problem Description: data set DataSet, target scale of data set dataset_{Scale_{target}}, hoping to get the target scale data meet minimum support degree and minimum reliability association rules by multi dimension association rules analysis.

Basic steps of algorithms:

1. select the benchmark scale Benchmark_{Scale}, determine the benchmark scale data set dataset_{BenchmarkScale}. Benefitting analysis as the principle, according to the available computing resources, select the scale which can present the maximum utility of the existing computing resources as the scale of the benchmark.
2. Analysis all benchmark data sets by the minimum support degree, obtain frequent item set, and take the number of frequent item set union as the frequent item set candidate set of target scale data set. This candidate item set can greatest reflects the circumstances which target scale data implied frequent item sets;
3. Due to either scaling up or scaling down, it is necessary to determine the similarity matrix between benchmark data sets, so this step is to calculate the similarity between benchmark data sets; from a statistical point of view, frequent item sets is a statistical result of data set on behalf of the distribution of data set with the characteristics of itself; in statistics, similarity

coefficient is mainly used to compare the similarity and dispersion in the finite sample set. The coefficient between the finite set is equal to the intersection of two sets of elements contained in the number and the number of elements contained in the union ratio, such as shown in formula (1). The actual study researchers have applied the coefficient to data analysis. This paper uses the similarity which is between benchmark scale data sets and frequent item sets to estimate the similarity between the original data sets.

$$\text{Similarity}(C,D)=|C\cap D|/|C\cup D| \quad (1)$$

Using formula (2) computing similarity coefficient:

$$\begin{aligned} M_{mn} &= \text{Similarity}(\text{Frequentitem}_m, \text{Frequentitem}_n) \\ &= |\text{Frequentitem}_m \cap \text{Frequentitem}_n| \\ &\quad / |\text{Frequentitem}_m \cup \text{Frequentitem}_n| \end{aligned} \quad (2)$$

4. Determine the analysis direction. The impact factor is benchmark dataset on the upper scale the data on the amount of data on the ratio reflects the influence degree, tendency of upper scale data sets such as division of population data of a certain area in national scale, while the Han population accounts for the vast majority of the region's population, then the "national" benchmark dataset of Chinese Population data on the overall population accounted for relatively large, equal in Han population in the region to the influence degree of the population as a whole in the analysis of population data, and Han population showed the characteristic of the data also reflects the overall population data trend. Bunches matrix elements mainly based quasi scale data set in the upper scale data quantity accounted for ratio, pushed down elements is both the similarity coefficient;
5. Determine weight matrix of the benchmark scale data sets to the target scale datasets;
6. Screening final frequent item sets of goal scale data sets and generate association rules. Final estimated frequent item sets support for all candidates in the same set of minimum support comparison, select frequent item set is not less than the set composed and produced in accordance with the minimum confidence association rules, pseudo codes for this algorithm are as follows:

III. ALGORITHM ANALYSIS

Compared with the classical data analysis algorithms, the advantage is that the proposed algorithm is only a reference scale analysis data sets. You can get a number of different levels of scale datasets behind implicit association rules. If you use the classic data analysis algorithms to achieve the same effect, you need multiple scales respectively data sets analysis, namely multiple analysis, which will cause a huge time and space overhead.

In addition, the proposed algorithm itself enforcement mechanisms to deal with the first reference scale data sets, and then the results of the multi-scale analysis are derived, if the idea of parallel computing applied this algorithm, the number of reference scale analysis parallel data sets, and then collect the

results of parallel processing scale up or scale down be converted and derivation, will have a more significant efficiency has increased significantly. This large data processing is very useful. Thus, the proposed algorithm is very suitable for parallel computing, parallel multi-scale data analysis to solve program is feasible and practical significance.

The multi-scale characteristic refers to the data set itself some attribute or attribute set is related to geographical space, or time, or other relative size and size range, said clear scale meaning. In fact, the main research of multi-scale data analysis is a multi-scale and multi scale data to achieve the conversion of knowledge. Based on the theory of multi-scale data reached a preliminary multi-scale data to achieve this goal, from this perspective, scale process with multi-scale characteristic data sets has a clear basis for partitioning results in the data set also has a clear meaning of the scale; and for similar IBMT10I4D100K data set so that the multi-scale characteristic is not very obviously the data set is divided into sequential scale, process and result of meaning is not very clear. We use multi-scale data analysis algorithm framework and specific association Rule analysis and realize the conversion of knowledge at multiple scales, from the point of view of algorithm analysis, algorithm implemented with the multi-scale characteristic of the data set, no matter from the process or results, both have more practical significance, especially the need for multiple criteria decision. Therefore, whether in theory or in practical algorithms, the theory of multi-scale data and multi-scale data analysis algorithm are more suitable to have multi scale characteristics of the data set.

IV. EXPERIMENTS

In this paper, the feasibility, accuracy and efficiency of the ARAMS algorithm are verified by using the Z province data set and IBMT10I4D100K data set. Z province full population data set is a complete record of the population management and household registration and other spatial attribute information; geographical attributes can form the concept of stratification, has a good multi scale characteristics. The running environment of the experiment is Intel i7 CPU 3.40GHz, 8G memory, Windows 8 operating system, ORACLE 11g database system, using Octave implementation algorithm. This paper uses the ARAMS algorithm to analysis the benchmark data set by adopting the classical Apriori algorithm.

Compared with the experimental results of population data set figure 1 and Figure 2, it can be found that the accuracy of the ARAMS algorithm for scaling up operation is better than the scaling down, but the accuracy of the scaling up and scaling down parts are relatively high. From Figure 1 (a), (b) shows that the upper part of the coverage and accuracy are more than 90%, and in some cases even reached 100%. From Figure 2 (a), (b) reflected in the coverage and accuracy is slightly worse than on the scaling up, but also is more than 80%. Above experiment results verify the accuracy and feasibility of the algorithm. It shows that the ARAMS algorithm is able to obtain the real frequent itemsets of target scale data set from the frequent item sets contained in the benchmark data set. There is a steep drop phenomenon in Figure 1 (a), (b) when the support is about 20%. It is due to the Y shaft size is very fine, so the subtle changes of y value in the figure will show a large change. In fact, it only

changed 5%. Figure 2 (b) shows that with the increase of the minimum support degree, the accuracy is on the upward trend, which is caused by the decrease of the proportion of false positive and false negative term sets in the overall result of frequent items. In Figure 1 (c) and Figure 2 (c) show that the average estimation error is low, especially on the experimental results of the scaling up algorithm, illustrate that the ARAMS algorithm has a good performance in the estimation of the support. In the aspect of execution efficiency, we can see from figure 1 (d), the advantage of proposed algorithms is more obvious than Apriori. From Figure 2 (d), it can be seen that the proposed algorithm has higher efficiency.

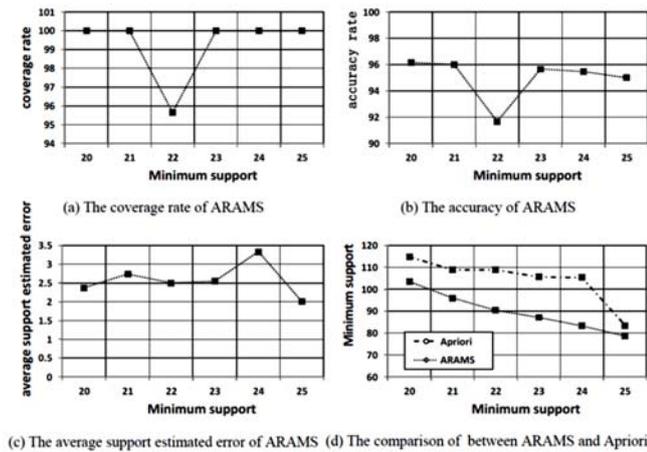


Figure 1. The scaling up experimental results of ARAMS

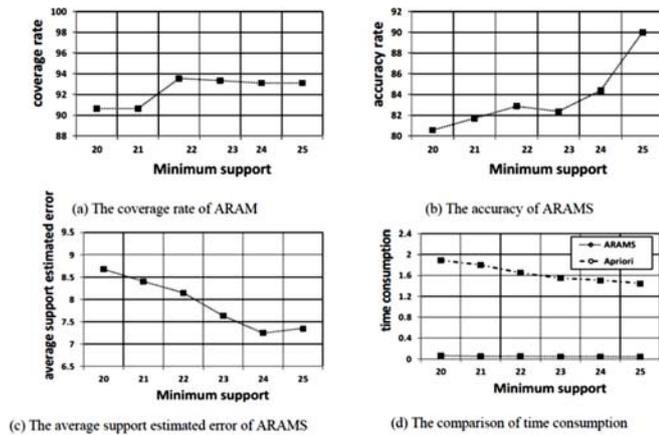


Figure 2. The scaling down experimental results of ARAMS

V. CONCLUSION

This paper introduces the basic idea of multi-scale space theory to the field of data analysis. This paper proposes data scale partitioning and data scale definition based on concept hierarchy, present the relationship between the upper and lower of scale data, and establish the theory foundation for the multi scale data space. This paper also proposes the definition and classification for multi scale data analysis, proposes algorithm framework for multi scale data analysis and present the association rule analysis algorithm for multi scale big data. The

algorithm uses benchmark data sets analysis result and the influence weight of benchmark scale data set to the target scale data set to derive the association rules behind object scale data set, realize knowledge across scales derived and provides the possibility for multiple criteria decision. Finally, we conduct experiments to verify the accuracy and efficiency of the proposed algorithm. The experimental results show that the algorithm has a high coverage and accuracy, and has a low support degree estimation error, and the efficiency is higher than the traditional Apriori algorithm.

The next step, we will focus on the following aspects: research on multi-scale data analysis scaling up and scaling down, analysis the variation rule of cumulative error and accuracy in the case of cross scale; research more perfect multi-scale data theory system; multi-scale data analysis algorithm framework is applied in such as classification, clustering and other data analysis application, explore the multiscale classification and clustering analysis algorithm, and improve these experiments of the algorithms; further improve the coverage rate, accuracy degree and efficiency of the multiscale association rule analysis algorithm in the practical application, also explore better weight coefficient calculation, reduces the estimation error of support degree from theory and practice.

ACKNOWLEDGMENT

This research is supported in part by Zhejiang Provincial Natural Science Foundation of China (No. LY16F020012), Ningbo Key Laboratory of intelligent home appliances (No. 2016A22008), Key Research Center of Philosophy and Social Science of Zhejiang Province-Modern Port Service Industry and Creative Culture Research Center, Zhejiang Province Public Technology Research and Industrial Project (No. 2015C32124), Projects in Science and Technique of Ningbo Municipal (No. 2014C10047 and No. 2012B82003), National Natural Science Foundation of China (No.61300202) and Natural Science Foundation of Shanghai (No. 13ZR1452900). Zheng Xu is the corresponding author.

REFERENCES

- [1] Belkin, M. and P. Niyogi, Laplacian eigenmaps for dimensionality reduction and data representation. *Neural computation*, 2003. 15(6): p. 1373-1396.
- [2] Hu, C., et al., Semantic link network-based model for organizing multimedia big data. *Emerging Topics in Computing, IEEE Transactions on*, 2014. 2(3): p. 376-387.
- [3] Xu, Z., et al., Semantic based representing and organizing surveillance big data using video structural description technology. *Journal of Systems and Software*, 2015. 102: p. 217-225.
- [4] Bell, R.M. and Y. Koren, Lessons from the Netflix prize challenge. *ACM SIGKDD Explorations Newsletter*, 2007. 9(2): p. 75-79.
- [5] Hu, C., et al., Video structural description technology for the new generation video surveillance systems. *Frontiers of Computer Science*, 2015. 9(6): p. 980-989.
- [6] Xu, Z., et al., Crowdsourcing based social media data analysis of urban emergency events. *Multimedia Tools and Applications*, 2015: p. 1-18.
- [7] Bellazzi, R., et al., Temporal data mining for the quality assessment of hemodialysis services. *Artificial intelligence in medicine*, 2005. 34(1): p. 25-39.
- [8] Luo, X., et al., Building association link network for semantic link on web resources. *Automation Science and Engineering, IEEE Transactions on*, 2011. 8(3): p. 482-494.

- [9] Xu, Z., et al., Crowdsourcing based description of urban emergency events using social media big data. 2016.
- [10] Blondel, V.D., et al., Fast unfolding of communities in large networks. *Journal of statistical mechanics: theory and experiment*, 2008. 2008(10): p. P10008.
- [11] Xu, Z., et al., Generating temporal semantic context of concepts using web search engines. *Journal of Network and Computer Applications*, 2014. 43: p. 42-55.
- [12] Xu, Z., et al., Semantic enhanced cloud environment for surveillance data management using video structural description. *Computing*, 2016. 98(1-2): p. 35-54.
- [13] Burnett, C. and T. Blaschke, A multi-scale segmentation/object relationship modelling methodology for landscape analysis. *Ecological modelling*, 2003. 168(3): p. 233-249.
- [14] Ding, H., et al., Querying and mining of time series data: experimental comparison of representations and distance measures. *Proceedings of the VLDB Endowment*, 2008. 1(2): p. 1542-1552.
- [15] Džeroski, S., Multi-relational data mining: an introduction. *ACM SIGKDD Explorations Newsletter*, 2003. 5(1): p. 1-16.
- [16] Huck, K.A. and A.D. Malony, Perfexplorer: A performance data mining framework for large-scale parallel computing. in *Proceedings of the 2005 ACM/IEEE conference on Supercomputing*. 2005. IEEE Computer Society.
- [17] Jenatton, R., et al. Multi-scale mining of fMRI data with hierarchical structured sparsity. in *Pattern Recognition in NeuroImaging (PRNI)*, 2011 International Workshop on. 2011. IEEE.
- [18] Khan, S.S. and A. Ahmad, Cluster center initialization algorithm for K-means clustering. *Pattern recognition letters*, 2004. 25(11): p. 1293-1302.
- [19] Knobbe, A., et al., Multi-relational data mining. 1999.
- [20] Xu, Z., et al., Knowle: a semantic link network based system for organizing large scale online news events. *Future Generation Computer Systems*, 2015. 43: p. 40-50.
- [21] Kopanas, I., N.M. Avouris, and S. Daskalaki, The role of domain knowledge in a large scale data mining project, in *Methods and Applications of Artificial Intelligence*. 2002, Springer. p. 288-299.
- [22] Lancaster, A., et al. PyPop: a software framework for population genomics: analyzing large-scale multi-locus genotype data. in *Pacific Symposium on Biocomputing*. Pacific Symposium on Biocomputing. 2003. NIH Public Access.
- [23] Li, S.-T., S.-W. Chou, and J.-J. Pan. Multi-resolution spatio-temporal data mining for the study of air pollutant regionalization. in *System Sciences, 2000. Proceedings of the 33rd Annual Hawaii International Conference on*. 2000. IEEE.
- [24] Li, S.-T. and L.-Y. Shue, Data mining to aid policy making in air pollution management. *Expert Systems with Applications*, 2004. 27(3): p. 331-340.
- [25] Low, Y., et al., Distributed GraphLab: a framework for machine learning and data mining in the cloud. *Proceedings of the VLDB Endowment*, 2012. 5(8): p. 716-727.
- [26] Machiraju, R., et al., EVITA—Efficient Visualization and Interrogation of Tera-Scale Data, in *Data mining for scientific and engineering applications*. 2001, Springer. p. 257-279.
- [27] Mennis, J. and D. Guo, Spatial data mining and geographic knowledge discovery—An introduction. *Computers, Environment and Urban Systems*, 2009. 33(6): p. 403-408.
- [28] Streilein, W., et al. Fused multi-sensor image mining for feature foundation data. in *Information Fusion, 2000. FUSION 2000. Proceedings of the Third International Conference on*. 2000. IEEE.
- [29] Tsytsarau, M. and T. Palpanas, Survey on mining subjective data on the web. *Data Mining and Knowledge Discovery*, 2012. 24(3): p. 478-514.
- [30] Vespignani, A., Predicting the behavior of techno-social systems. *Science*, 2009. 325(5939): p. 425.

Distribution and Continuity of Developers' Contributions in OSS Projects: A Case Study

Zhongjie Wang¹, Dewayne E. Perry², Xiaofei Xu¹

¹ Harbin Institute of Technology, Harbin, China 150001

² The University of Texas at Austin, Austin, TX, USA 78712

rainy@hit.edu.cn, perry@ece.utexas.edu, xiaofei@hit.edu.cn

Abstract– Open Source Software (OSS) is usually developed by geographically-distributed developers in a collaborative manner. Different developers exhibit different behaviors and make diversified contributions to OSS projects. Objective of this paper is to discover individualized characteristics and common patterns of how developers contribute to OSS projects. Continuity is used to delineate how a developer actively contributes to the project over time. Case studies on two OSS projects reveal some significant phenomena on the distribution of developers' contributions relative to absolute time and relative to the milestones (i.e., releases) of OSS projects. We have found that OSS developers' contributions exhibit the “temporal locality”, and most of the releases of an OSS project are dominated by the contributions of a limited number of developers.

Keywords– Open Source Software; social development; developers' contributions; continuity; temporal distribution

1. Introduction

Different developers have different social and technical backgrounds. They joined an OSS project with different motivations [1, 2]. Consequently, the degree of their participation and the level of their contributions are diversified [3]. A set of individualized characteristics on the behaviors of OSS developers that distinguish one from others who belong to the same OSS project team do indeed exist [4].

Most of the recognized characteristics are closely related to externally visible behaviors that developers exhibit in the history of OSS projects. This paper is focused on exploring the personalized contributing styles of developers from a temporal dimension. A developer initiatively enforces to make contributions to OSS projects, especially code commits. Targeting at the contributions, a metric called “continuity” which delineates how frequently and consecutively a developer makes contributions to a project and is measured by the times, contributed Line of Code (LoC), and temporal distribution of his active behaviors, is studied. We introduce an approach of identifying and measuring the continuity of contributions from public data

in OSS repositories, in which the continuity is refined into a set of fine-grained metrics based on the temporal distribution of the active behaviors of each developer, with respect to the absolute timeline (defined by calendar time) and the relative timeline (defined by a series of releases, or milestones, of an OSS project). Entropy is employed to give an overall measurement of the continuity. Case studies are conducted on two OSS projects (JUnit and Guava) hosted on GitHub. The identified characteristics are visualized, and developers are compared in terms of these characteristics to discover the difference and similarities among their behaviors.

This study tries to answer three research questions:

RQ1: How are a developer's contributions distributed on the absolute timeline?

RQ2: How are a developer's contributions distributed relative to the milestones of the OSS project?

RQ3: Are there distinct characteristics of different developers on the distribution/continuity of their contributions?

By case studies we find that, (1) Contributions of a developer often show temporal locality, and there are some long intervals that split a developer's contributions into stages; the more contributions a developer makes, the more evenly his contributions are distributed over time, with averagely shorter intervals; (2) Most of the releases of an OSS project are dominated by contributions of a limited number of developers, respectively, i.e., the active behaviors of a few developers contribute most of the changes of a release relative to its previous one. These would help OSS project coordinators get a good understanding on the working habits of developers in the project team, thus facilitate proactive and accurate project management activities such as task allocations.

The remainder is organized as follows. Candidate projects and the independent/dependent variables of the case study are presented in Section 2. Section 3 is the study approaches and results, followed by threats of validity in Section 4. Section 5 and 6 are the related work and conclusions.

2. Study Setup

2.1. Candidate OSS Projects

Two OSS projects from GitHub, JUnit¹ and Guava², are selected for the case study. They are diversified in many perspectives. From the perspective of time to live, JUnit is much older for above 14 years and Guava is for nearly 6 years. From the perspective of scale, Guava is larger with more than 1,700 source files and above 400K LoC, and JUnit is a medium project with 300-400 files and 40K LoC. From the perspective of team size, JUnit has a larger team with more than 130 developers, while Guava has a relatively smaller team. From the perspective of behavior intensity, Guava has the larger number of commits than JUnit, and on average, although Guava has a smaller team, its developers seem more active and commit more average LoC than JUnit.

2.2. Variable Selection

2.2.1 Independent Variables (IVs)

An OSS project has a team T and a sequence of code commits CS . Every time a developer submits code to the project repository, does he carry out an active behavior explicitly exhibited as the expansion or modification of existing source code in two forms: addition and deletion.

A commit $c \in CS$ is defined as $c = \langle PC, F, d, t \rangle$, where PC is c 's parent commits on which c 's code changes are based, and each commit might have 0, 1 or multiple parent commits; F is the files contained in c , and $\forall f \in F, l^+(f), l^-(f)$ are the code lines added into and deleted from f , respectively, compared with the corresponding files in PC ; d is the developer who authored the code changes in c ; and t is the commit time of c .

Commits of a project form a sequence in which they are sorted by commit time in chronological order. In terms of a developer d_i , all the commits authored by d_i form a sub-sequence of CS and is denoted by CS_i . Multiple developers alternately check out codes from repository and commit changes back, so their commit sequences are interwoven with each other. Denote $\varphi_i = |CS_i|$, and $\sum_{i=1}^{\tau} \varphi_i = |CS|$.

In GitHub, a set of commits could be grouped together as a release being a phase achievement of the project³. Here we simply define a release r_j as a set of commits (the number of commits is $|r_j|$). Suppose there are totally γ releases in a project and they form a sequence $\langle r_1, \dots, r_\gamma \rangle$ in terms of the release date.

A set of git commands are used to collect above-mentioned variables from git repositories,

¹<https://github.com/junit-team/junit>

²<https://github.com/google/guava>

³<https://help.github.com/articles/creating-releases/>

such as `git rev-list`, `git diff`, `git show`, `git for-each-ref --refs/tags''`, etc.

2.2.2 Dependent Variables (DVs)

The dependent variables (DVs) are constructed in terms of two timelines: (1) The absolute timeline defined by calendar time; (2) The relative timeline defined by the date of releases;

For the former, the lifespan of a project is split into N number of timeslots $\langle p_1, \dots, p_N \rangle$. The length of a timeslot may be 1 day, 1 week, 1 month, or any length of time. The following are the DVs used under the absolute timeline:

(1) $V_{times}(d_i), V_{loc}(d_i)$: a developer d_i 's distribution vectors of the times and LoC of active contributing behaviors in N timeslots, i.e., $V_{times}(d_i) = \langle \varphi_{i1}, \varphi_{i2}, \dots, \varphi_{iN} \rangle$, $V_{loc}(d_i) = \langle l_{i1}^+, l_{i2}^+, \dots, l_{iN}^+ \rangle$.

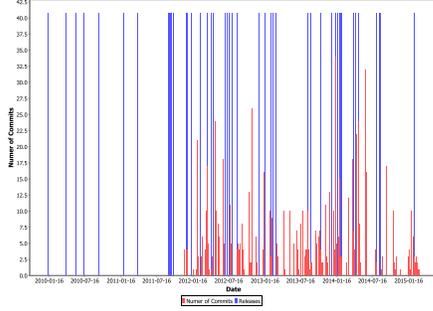
We define $CS_{ik} = \{c_j | c_j \in CS_i \wedge t(c_j) \in p_k\}$ as the commits that belong to CS_i (the commit sequence of d_i) and fall into the timeslot $p_k (1 \leq k \leq N)$. Then, $\varphi_{ik} = |CS_{ik}|$ is d_i 's times of active behaviors occurring during p_k ; $l_{ik}^+ = \sum_{c_j \in CS_{ik}} l^+(c_j)$ is the total LoC that d_i contributes during p_k , in which $l^+(c_j) = \sum_{f \in F(c_j)} l^+(f)$ is the sum of "additions" contained in the files of c_j , indicating the d_i 's total contribution made in the commit c_j .

(2) $V_{interval}(d_i)$: the vector of intervals between two consecutive active behaviors of d_i , i.e., $V_{interval}(d_i) = \langle \delta_{i1}, \dots, \delta_{i(\varphi_i-1)} \rangle$ where $\delta_{ij} = t(c_{i(j+1)}) - t(c_{ij})$ measures the time interval between c_{ij} and $c_{i(j+1)}$ both included in d_i 's commit sequence CS_i .

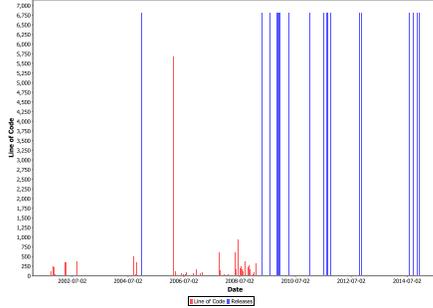
(3) $H_{times}(d_i), H_{loc}(d_i)$: the continuity entropy of active contributing behaviors of a developer d_i . Entropy has been widely applied in software engineering for various measurement on software entities and software development activities, such as software complexity entropy [5] and software change entropy [6]. Here we borrow this concept to measure how a developer's contributions on source code are distributed over time. The first entropy is "times entropy" measuring the uncertainty in the times distribution of d_i 's active behaviors in N timeslots $\langle p_1, \dots, p_N \rangle$, i.e., $H_{times}(d_i) = -\sum_{k=1}^N (\frac{\varphi_{ik}}{\varphi_i}) \log(\frac{\varphi_{ik}}{\varphi_i})$. The second one is "LoC entropy" which measures the continuity of the LoC in d_i 's active behaviors, i.e., $H_{loc}(d_i) = -\sum_{k=1}^N (\frac{l_{ik}^+}{l_i^+}) \log(\frac{l_{ik}^+}{l_i^+})$ where $l_i^+ = \sum_{c_j \in CS_i} l^+(c_j)$ is the total LoC that d_i has contributed during the lifespan of the project.

For relative timeline, $V(r_k)$ delineates the relative distribution characteristics w.r.t. the releases of a project:

(4) $V(r_k)$: the vector of the contribution ratios of all the developers in one release period r_k , i.e., $V(r_k) = \langle \sigma_{1k}, \sigma_{2k}, \dots, \sigma_{\tau k} \rangle$ where σ_{ik} is the ratio of d_i 's contribution (LoC) relative to the total contributions that all developers have made during the period from the release r_{k-1} to r_k , i.e., $\sigma_{ik} = \frac{l_i^+(r_k)}{l^+(r_k)}$, and $l_i^+(r_k) = \sum_{c_j \in r_k \wedge c_j \in CS_i} l^+(c_j)$, $l^+(r_k) = \sum_{c_j \in r_k} l^+(c_j)$, $\sum_{k=1}^{\tau} \sigma_{ik} = 1$, and $\tau = |T|$ is



(a) $V_{times}(d_1)$ of Guava.



(b) $V_{loc}(d_2)$ of JUnit.

Figure 1. Barcode visualizing contribution distribution w.r.t. absolute timeline

the total number of developers.

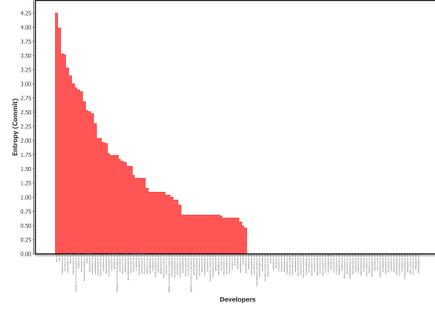
3. Study Results

3.1. Contribution Distribution w.r.t. Absolute Time

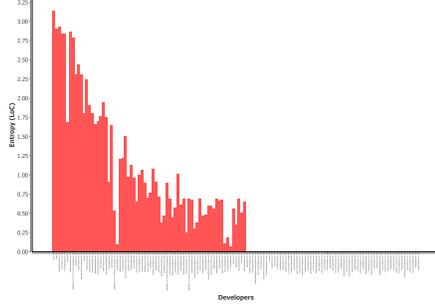
To exhibit the distribution of developers’ contribution w.r.t. the absolute timeline, “barcode” is employed to visualize the distribution vectors $V_{times}(d_i)$ and $V_{loc}(d_i)$ for each developer d_i .

Fig. 1 gives the barcode of two developers from JUnit and Guava, respectively. Fig. 1(a) is the times distributions of a Guava developer, and Fig. 1(b) is the LoC distribution of a JUnit developer. The timeline is split into N timeslots (the length of timeslot is 1 day; $N=5,235$ for JUnit and $N=2,003$ for Guava). The red vertical bars are active contributions, and the height of the bars shows the times or LoC of behaviors occurring in the specific timeslot. As a reference, blue bars represent the releases of the project.

Barcode is a straightforward way to visualize the temporal distribution of developers behaviors. Without more precise quantitative metrics, the continuity and the intensity of active behaviors at different times can be intuitively observed from the barcode. For example, d_1 has intensive and continuous active behaviors from the middle of the project, d_2 ’s active behaviors are split by two long intervals, and d_3 ’s behaviors are



(a) H_{times} of JUnit.



(b) H_{loc} of JUnit.

Figure 2. Comparison of the continuity entropy among developers

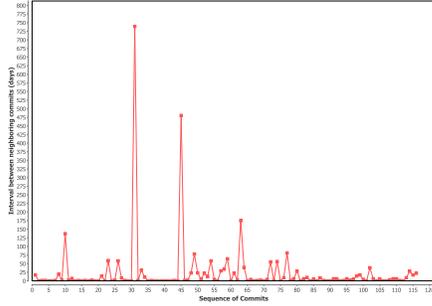
concentrated in a very short period.

By observation of the barcode for 133 developers in JUnit and 44 in Guava, we find that the shapes of barcode of different developers are quite diversified. Result of Chi-squared test of independence has shown that there are significant difference between the distribution vectors of any pair of developers belonging to the same project. It confirms that the distribution vectors of active behaviors can be used as individualized characteristics of OSS developers.

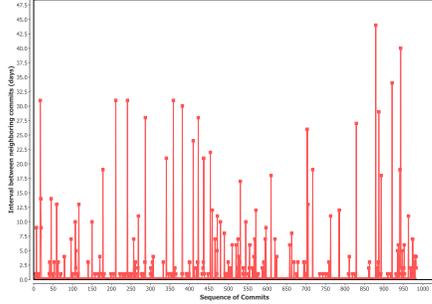
Fig. 2 demonstrates the continuity entropy H_{times} and H_{loc} of developers in JUnit. Developers are sorted by H_{times} in descending order. A higher entropy indicates that the developer’s contributions are distributed over time in a more balanced way. The entropy of some developers in the right equals to 0 because all their active behaviors concentrate in only one timeslot.

From the comparisons between H_{times} and H_{loc} for JUnit, and between the ones for Guava, the distributions of H_{times} and H_{loc} show almost the same shape. The Spearman’s rank-order correlation test shows that the two entropy are highly correlated. For JUnit, the correlation coefficient is 0.863 ($p < 0.01$); for Guava, it is 0.907 ($p < 0.01$).

Nevertheless, there are still some inconsistencies between them. For example, a developer has higher H_{times} while his H_{loc} is comparatively smaller. This implies that, the distribution of LoC in his active behaviors is more unbalanced and



(a) Developer 3 of JUnit.



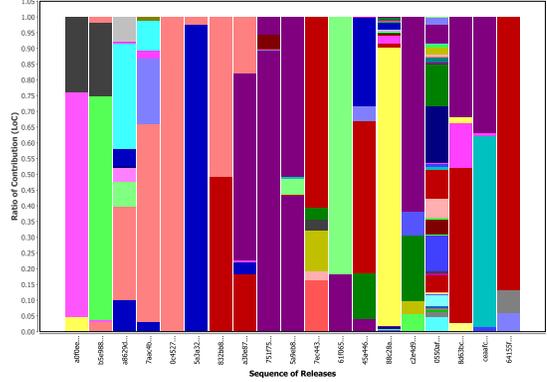
(b) Developer 4 of Guava.

Figure 3. Distribution of intervals between neighboring contributing behaviors

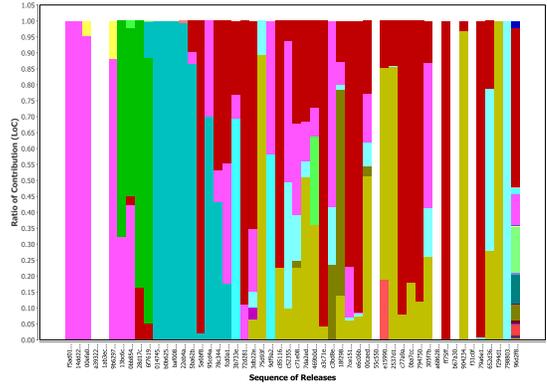
changes more drastically than the distribution of behavior times. For example, there is a developer in `JUnit` with almost all his LoC (nearly 45 KLoC) contributed in the last period of his participation, and the LoC in other periods are very small, thus his H_{times} is in rank 24 among all developers while his H_{loc} is in rank 70 only.

We observe from the barcode that, contributions of many developers are distributed in a stage-wise manner, i.e., between two phases of intensive active behaviors, there is usually a long interval. Fig. 3 shows $V_{interval}$ of two developers in `JUnit` and `Guava`, respectively. Although a majority of the intervals are close to x -axis (indicating the intervals are short and the contributions are approximately consecutive), there are obviously some outliers that have relatively higher values and represent the long gaps between stages of behaviors. For example, the developer in Fig. 3(a) has two obvious intervals, the first one is one year, and the second is two years. By contrast, although Fig. 3(b) seems to have more frequent fluctuations, the amplitudes of these fluctuations are actually smaller than the ones of the first developer (see the different scale of y -axis in the two figures).

Summary: (1) Barcode is an intuitive method for visualizing temporal distribution of a developer’s contributions, and entropy can measure the continuity of contribution distribution; (2) There are temporal locality in the contributions of developers, i.e., most of the intervals between neighboring behaviors are



(a) JUnit.



(b) Guava.

Figure 4. Stacked barchart of the relative contribution ratio of developers in releases

short; but there are a small number of long intervals that split the behavior sequence into stages.

3.2. Distribution of Contributions w.r.t. Releases

$V(r_k) = \langle \sigma_{1k}, \sigma_{2k}, \dots, \sigma_{\tau k} \rangle$, vector of the contribution ratios of developers in one release period r_k , is visualized first. In the form of stacked barchart in Fig. 4, all the releases appear side by side for comparison. The x -axis is a set of releases sorted in chronological order, and the y -axis is the ratio of contributions. Sub-bars with the same color in different releases represent the same developer.

Some significant facts are observed: (1) There are usually a few developers whose contributions dominate one release. The number of dominant developers is so few that the contributions of other developers involving in the same release eventually become less obvious. (2) A small number of releases possess a large number of developers (e.g., 4th release from the end in `JUnit` and the last release in `Guava`). (3) Significant contributions that a developer makes are generally located in one single or multiple consecutive releases.

We use 5% as the threshold of judging whether a developer

is a “significant contributor” of a release, i.e., if the ratio of the added LoC of a developer relative to the total added LoC of a release is above 5%, he is considered to be a significant developer of this release. Sensitivity analysis has shown that the value of this threshold ranging from 2% to 10% yields similar results.

In JUnit, the range of number of contributors in one release is [1, 59], but most of releases have less than 10 developers, above 80% releases have less than 3 significant contributors, and the contribution ratio of the leading developer is usually far beyond the one of the runner-up. Similar situation can be found in Guava, too. Comparing the two projects we find that the average and median number of contributors of one release in JUnit are both larger than the ones in Guava (average: $9.42 > 3.30$; median: $5 > 2$), and the average variance of contribution ratios of these contributors in JUnit is smaller than the one of Guava ($0.025 < 0.04$). These statistical results show that the participation degree of developers in JUnit is higher than the one in Guava; in other words, developers in JUnit are more active than the ones in Guava.

In the heat chart of Fig. 5, if a developer in y -axis is a significant contributor of a release in x -axis, the corresponding grid is filled with red color. Developers who are not significant contributors of any releases are not included in the figure. The heat chart further demonstrates the continuity (temporal locality) of the active behaviors of developers relative to the milestones of OSS projects, i.e., for most of developers, their active behaviors dominate one or several releases which are adjacent or close to each other.

To note that, the absolute timespan of different release periods are quite different (see the distance between blue vertical bars in Fig. 1), and the total LoC of different releases are greatly different, too; thus, even if the contributions of a developer dominate a release, it does not indicate that his absolute LoC contributions are large. Once again, what Fig. 4 shows are the ratio of contributions in each release period.

To sum up, the conclusions drawn in this section are: (1) Being the milestones of an OSS project, a majority of releases are dominated by very few developers whose active behaviors constitute the main code changes of the releases. The number of contributors of most releases is usually low, and only very few releases have a large number of contributors whose contributions are relatively balanced. (2) Contributions of a developer tend to be distributed in neighboring or near releases. This demonstrates the temporal locality of a developer’s active behaviors relatively to the staged milestones of the project. (3) The distribution of the number of dominating releases of all the developers exhibits approximate power law.

4. Threats to Validity

We select DVs in three levels, i.e., the temporal distribution of contributions for the most detailed level; the intensity and

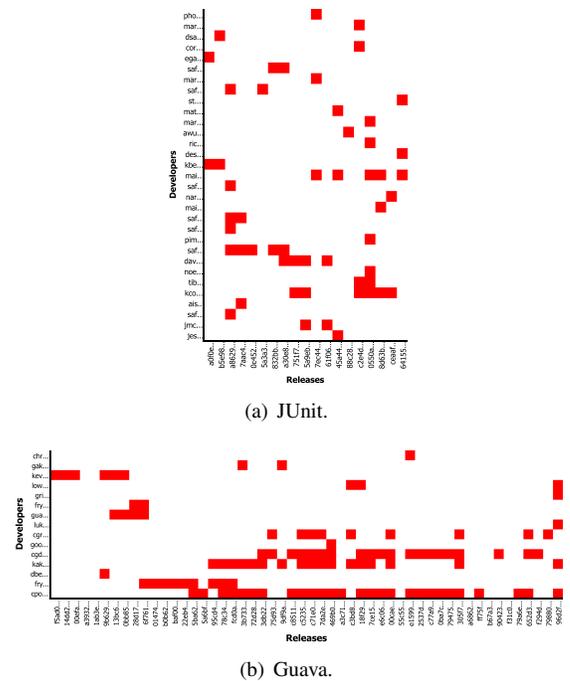


Figure 5. Heat chart of significant developers relative to releases

contribution ratio relative to project releases for the middle level; and the entropy for the global level. These DVs are calculated from the IVs which are derived from the open data of the OSS repositories hosted on Github. Such multi-level DVs ensure the comprehensiveness and understandability of the measurement.

Active behaviors are defined as the contributing activities that developers take on their own initiatives, mainly the committing activities that have explicit code contributions to projects. The exclusion of other types of activities (such as forking a project, commenting, or reporting an issue) from the study would result in some incompleteness of the identified individualized behavioral characteristics. Nevertheless, many existing research adopted similar approaches as ours.

Only the added lines of code in commits are considered but the “deletions” are ignored. However, if a developer removes redundant or buggy code without adding new lines, he still has contributions to the project. It makes the measurement on the continuity of contributions a little biased.

Two OSS projects are selected for the case studies. In an attempt to address the generalizability of our findings, more studies conducted on larger-scale and longer-living OSS projects are necessary.

5. Related Work

Code commitment is considered as the dominant behaviors of developers because they take direct effect on the source code

of OSS projects, e.g., Teyton *et al* [7] extracted a developer's contribution by delta analysis on source code changes being the result of developer behaviors. Yang *et al* [8] analyzed the commit activities of developers and found that they are not regularly distributed along with time, thus "barcode" is used to visualize the distribution of commit sequence of each developer. Our study extends their barcode by (1) using the height instead of the width of bars to signify the attributes (times, LoC, and intensity) of behaviors, and (2) putting the barcode into a timeline, so that the temporal distribution of contributions is visualized more precisely.

Siy *et al* [9] presented a method to summarize work history of developers in terms of the files they have modified over time using the time series segmentation technique, and the segmentation result is used as an individualized feature of developers and some evolution patterns of developer behaviors are found. In our study, we use two types of time segmentation approaches to split the developer behaviors into segments, i.e., absolute and relative timelines.

Lin *et al* [10] studied the relationship between the distribution of commit behaviors and the releases of OSS projects and identified five distinct zones in the distribution of commit activities across various releases; their conclusion indicates that developer behaviors are not independent but partially influenced by global constraints of projects such as "deadline". In our study, we borrow this idea and use releases as the relative timeline to study the correlation between developers' contributions and project milestones.

Weissgerber *et al* [11] used a transaction visualization technique to show commit sequences in a project and used file-author matrix to visualize evolution history of a file in terms of developers. This approach stands on the file's point of view; by contrast, we use a line of code as basic change unit rather than a large-grained file, thus the result would be more accurate.

6. Conclusion

We conduct an case study on the individualized features and common patterns of contributions of OSS developers. Continuity of active contributing behaviors is focused to explore how developer behaviors are distributed over time by a set of fine-grained metrics. The conclusions are: (1) Active behaviors of a developer often show temporal locality. Some long intervals split a developer's behaviors into stages, and the more contributions a developer makes, the more evenly his active behaviors are distributed over time, with averagely shorter intervals between neighboring stages of intensive active behaviors. (2) Most of releases of a project are dominated by limited number of developers, respectively, and the total number of contributors in each release is usually low. Active behaviors of a developer tend to be distributed in neighboring or near releases.

The findings of this study would help OSS project coordinators get deep insight in the behavioral characteristics of team

members so as to improve their project management practices. Future work will be conducted on this perspective.

Acknowledgment

Work in this paper is supported by the Natural Science Foundation of China (No. 61272187, 61472106).

References

- [1] J. A. Roberts, I.-H. Hann, and S. A. Slaughter, "Understanding the motivations, participation, and performance of open source software developers: A longitudinal study of the apache projects," *Management Science*, vol. 52, no. 7, pp. 984–999, 2006.
- [2] G. Hertel, S. Niedner, and S. Herrmann, "Motivation of software developers in open source projects: an internet-based survey of contributors to the linux kernel," *Research Policy*, vol. 32, no. 7, pp. 1159–1177, 2003.
- [3] B. J. Dempsey, D. Weiss, P. Jones, and J. Greenberg, "Who is an open source software developer?" *Comm. the ACM*, vol. 45, no. 2, pp. 67–72, 2002.
- [4] M. Y. Allaho and W.-C. Lee, "Trends and behavior of developers in open collaborative software projects," in *2014 Int'l Conf. Behavior, Economic and Social Computing*. IEEE, 2014, pp. 1–7.
- [5] W. Harrison, "An entropy-based measure of software complexity," *IEEE Trans. Software Engineering*, vol. 18, no. 11, pp. 1025–1029, 1992.
- [6] G. Canfora, L. Cerulo, M. Cimitile, and M. Di Penta, "How changes affect software entropy: an empirical study," *Empirical Software Engineering*, vol. 19, no. 1, pp. 1–38, 2014.
- [7] C. Teyton, M. Palyart, J.-R. Falleri, F. Morandat, and X. Blanc, "Automatic extraction of developer expertise," in *18th Int'l Conf. Evaluation and Assessment in Software Engineering*. ACM, 2014, p. 8.
- [8] W. Yang, B. Shen, and B. Xu, "Mining github: Why commit stops—exploring the relationship between developer's commit pattern and file version evolution," in *APSEC'13*. IEEE, 2013, pp. 165–169.
- [9] H. Siy, P. Chundi, and M. Subramaniam, "Summarizing developer work history using time series segmentation: challenge report," in *MSR'08*. ACM, 2008, pp. 137–140.
- [10] S. Lin, Y. Ma, and J. Chen, "Empirical evidence on developer's commit activity for open-source software projects," in *SEKE'13*, 2013, pp. 455–460.
- [11] P. Weissgerber, M. Pohl, and M. Burch, "Visual data mining in software archives to detect how developers work together," in *MSR'07*. IEEE, 2007, pp. 9–16.

A Two Phase Case Study on Implementation of Open Source Development Practices within a Company Setting

Alma Oručević-Alagić, Martin Höst
Department of Computer Science, Lund University, Sweden
(alma.orucevic-alagic, martin.host)@cs.lth.se

Abstract

Implementation of open source development practices within commercial settings can bring benefits such as improved source code quality, lower maintenance costs, and increased innovation. However, a widespread in-house implementation of the practices has not been observed. The goal of this research is to understand factors which hinder the implementation. For the purpose, development practices of a large, global software and hardware organization that bases its products on open source software, and has over a decade long experience of contributing to various open source projects were studied. The results were validated through a set of structured interviews and a focus group meeting. It is found that the initial implementation of the process has not been carried out in a planned and systematic way within the company. The results of the follow-up focus group meeting show that while the company's practices acquired a higher degree of alignment over a two-year period, the change was necessitated by a need to have a more efficient development effort across new, globally distributed, development sites.

1. Introduction

Open source software (OSS) has influenced the way software is produced and distributed. Some companies use open source as another type of off-the-shelf software, while others integrate it into their software products and actively participate and contribute code to open source communities, see for example Höst and Oručević-Alagić [6]. By participating in OSS development processes, companies gain experience in how online, distributed collaboration effort is organized, and how information and knowledge are managed in these projects. While the software built by OSS communities spans a wide range of domains, sizes, and maturity levels, an increased industry interest and involvement

with OSS came with the emergence of a large, complex, and industry-grade software products, such as Linux, Android, and Hadoop [19].

Most common profile of the developers contributing to OSS projects is that of unpaid, geographically distributed, and well integrated into highly organized and structured fabric of OSS projects [13]. However, with increased usage and participation of the industry in OSS projects, a greater participation of paid developers can be noted, and as OSS gains mainstream acceptance, business models and industry involvement strategies mature as presented e.g. by Fitzgerald [2]. However, understanding how OSS development practices could be applied in-house to enhance proprietary software development process requires further investigation for several reasons. Scacchi [16] argues that OSS development is an interesting alternative approach to development of large systems and suggests that further research, especially using empirical examination, is conducted in order to better understand OSS development practices (OSDP). A description on adopting open source development practices within the organizations, also known as inner source, was proposed by Stol and Fitzgerald [18]. They show three important aspects of software projects that are run as inner source: types of projects, practices and tools, and people and management. While there exist a number of case studies showing successful adoption of OSS in-house, HP [11], Lucent [4], and Nokia [10], more evidence is still needed to better understand how companies can apply OSDP.

In this case study we present alignment of software development practices and OSDP in a large, international, software and hardware company, referred to as the Case Company, that bases its products on open source software. The company has a long experience of working with mature communities, and has recognized the value of the OSDP. This study is a second part of the two phase study [12] which tracks the adoption of OSDP in commercial setting over a two year period.

The outline of this paper is as follows. In Section 2, the background information on OSDP is presented. In Section 3, the research approach is further defined. Section 4

presents the obtained results, while Section 5 discusses and analyses the obtained results in some more detail. Finally, conclusions are drawn in Section 6.

2. Background and related work

Examples of studies that demonstrate successful application of OSDP within commercial settings are the ones conducted in HP [11], Lucent [4], and Nokia [10]. The studies analyzed, for example, development of a complex software product across departments [4] using OSDP, and transitioning of an entire development to adopt OSDP [11]. The software produced in such manner is called “inner source”, “progressive open source”, or “closed open source”.

A case study conducted by Stol et al. [17] focuses on the challenges of building and integrating software products developed as a shared asset. In this case study the focus is on challenges of developing and integrating software developed as a shared asset within the company setting, and comparing these challenges with the challenges of integrating an open source software product developed outside of the company. The Stol et al. research [17] concludes that organizations can benefit in adoption of OSDP, but that more research in the area is needed to further identify and address the challenges of OSDP in company settings.

Melian and Mähring conducted a study in HP [11] observing the process of progressively transitioning HP’s development team to work under OSDP. The motivation for introduction of POS in HP is the business need to increase the development cost efficiency and shorten time to market by making software highly modular and reusable asset. The research has produced a comparative listing of open source and “progressive open source” development practices. Some of the biggest differences between the two practices lied in the aspects of organizational structure, time and budget to deliver, abundance of available human resources and reward system. They conclude that implementation of OSDP within a corporate setting can bring long lasting benefits in terms of development efficiency and code quality, but they also state that more research is needed to address differences in reward system, and control and monitoring of individual participants.

A case study conducted by Gurbani and Gavert in Lucent [4] provides another relevant insight into what happens when a software product is developed within a company as a shared asset, and employees from other departments are involved in its development through development process compliant with OSDP. The lessons learned from that case study are that source code ownership and the “many eyeballs” contributing to a transparent development process facilitate efficient software development especially if the software product is shared and highly utilized across different departments as it was the case in that case study [4].

All the studies referenced above ([11], [10], [17], [4]) identified the importance of having a common set of standard development tools, a single version control system, and a standardized change management system.

For the purpose of this study a set of Open Source development practices was identified based on the work by Fogel [3] and outlined in Table 1. The work details the most important aspects and characteristics of free software development, based on experiences gained with the Apache Subversion project [1], the OSS source code version control system with widespread use in open source and corporate setting. It provides a valuable insight on how the Subversion [1] community has been built and sustained over a period of twelve years. Besides analyzing the infrastructure needed to support the project in an online environment, Fogel [3] also elaborates on the importance of building a healthy environment culture, facilitating authority based on meritocracy and communication relying on standardized channels and formats.

While shared OSDP aspects, across different sizes and domains of mature OSS projects, include Infrastructure and Communication aspects, defined by IDs S1-S21, in Table 1, they can differ in governance types, defined by IDs S22-S24. For example, the Linux project adheres to the “benevolent dictator” management practice, where lieutenants are assigned for different parts of the code, but the ultimate decisions are made by Linus Torvalds. The community source governance type for library and related fields software, popularized by Kuali Open Library Environment [9], enables institutions to share development resources and influence development of a software project in a closed source setting, provided that in the later phase the project is open sourced. The government type applied in the Apache Subversion project is base on meritocracy, also referred popularly as “do-ocracy”, where roles, authority and promotion is based on the participants’ demonstrated knowledge and contributions to a project. We argue that such governance model can be suitable also for a closed source industry setting.

While the adoption of open Source practices can benefit companies [5], there are also some issues it raises. Some of the issues include development of products across organizational boundaries, especially in the companies where the development process is highly hierarchical.

3. Research methodology

3.1. Research approach

The research presented in this study is conducted as focus group meeting [14], [8] and it represents a second phase of the two-phase study as depicted in Figure 1, which depicts the entire research process. The participants of the

Table 1. OSS Framework

Aspect	Category	Subcategory	Id.
Infra-structure	Product Info	Features	S1
		Documentation	S2
		FAQ	S3
		News	S4
		Road Map	S5
		Security	S6
	Code Access	Download location	S7
		Binary package	S8
		Release Notes	S9
	Community Guide	Community Overview	S10
		Community Roles	S11
		Coding Conventions	S12
		Commit Conventions	S13
		Building and Testing	S14
		Debugging	S15
		Mailing Lists	S16
		Bugs/Issues	S17
		Releases	S18
Communication		Standardized	Message
	Channel		S20
	Norm		S21
Management	Meritocracy	Role	S22
		Promotion	S23
		Authority	S24

focus group meeting were the same individuals that were involved in the execution of the first phase of the study and thus were well acquainted with purpose and details of the study. The meeting discussion was structured around predefined interview questions with overall goal of assessing current level of OSDP implementation within the company and identifying and understanding factors behind any changes in the implementation.

The Case Company is a global market leader in software and hardware production within its field, and its core products are based on an OSS product licensed under GPL. The company has over a thousand employees and, through own and partners' offices, it is present in over 170 world countries. The Case Company is also a significant contributor to a number of different OSS communities.

The main questions that are analyzed in the study are:

RQ1 What differences in the level of implementation of OSDP presented in Table 1 can be noted over the past two years in the Case Company?

RQ2 What are the underlying reasons for the change in the OSDP implementation levels?

The main research question, RQ2, tries to understand and answer why certain practices are introduced and other practices are not introduced. In order to understand this, RQ1 focuses on what has happened at the case company during the last two years.

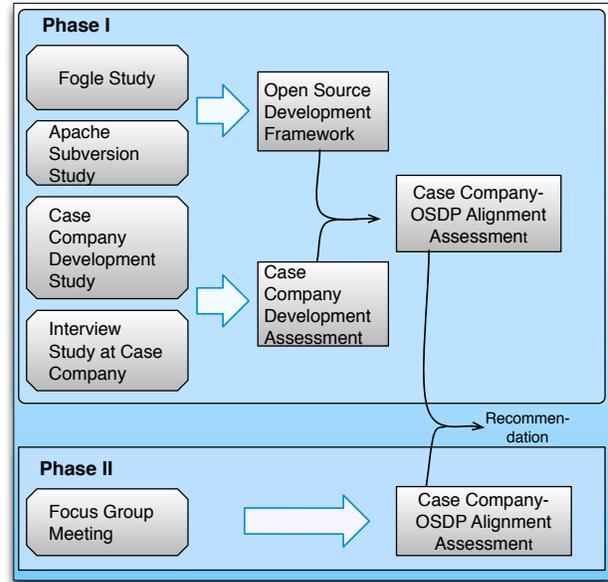


Figure 1. The two phase research process

In order to answer the questions it is necessary to understand the status before the two years started and what has happened during the two years, and why these changes have been made. Phase I (see Figure 1) of the study answers what practices were introduced two years before Phase II. Phase I, as presented in [12], was conducted by studying the alignment of the case company and Open Source practices, and to what extent developers thought that practices would be feasible to introduce in their environment. This was done by observing the case company and conducting interviews six persons with roles covering project management, technical lead, architect, and developer. Phase I was carried out over a period of 2 months with the first author visiting the case company during this time. Some further details about the conducted research are presented when the research validity is discussed in Section 3.2.

Phase II was, as described above, conducted through as a focus group meeting. In order to understand what has happened after Phase I, and by that being able to answer RQ1 and R2, a number of discussion questions were phrased and used at the focus group meeting:

1. How often do you use intra company online resources to discuss or solve project related tasks(e.g. online communication, knowledgebase, ongoing projects)?
2. How are change requests handled?
3. Can developers or smaller teams independently choose task to work on?
4. Which aspects of OSDP need to be implemented at

greater level and why?

In order to facilitate discussion in an efficient way, the following format was followed. For each discussion question participants were given some time to think about the question and write down their answers on an enclosed questionnaire. After formulating individual answers, each participant shared his answer with the group, which presented bases for a discussion. The individual answers were then collected by the researchers. During the discussion, one of the researchers also took notes.

The participants answers were transcribed, systematized based on commonality of their responses, and further analyzed by comparing them to the notes taken by one of the researchers. Based on this, a report was developed with summaries for each of the interview questions. During the analysis the results were also compared to the results from Phase I. The summaries are presented in Section 4 where they are also analyzed and compared to the results from the first phase of the study.

3.2. Validity

In this section the validity of the research is analyzed with respect to the types of validity threats typically present in qualitative studies: construct validity, internal validity, external validity, and reliability [14], [15]. As this study is a continuation of the two phased study, the validity assessment includes both of the phases.

The *construct validity* is concerned with the relationship between the subject of the study and what is measured, in this case the alignment of OSDP of mature OSS communities and software development practices of the Case Company. In order to properly identify mature Open Source development practices, the work by Fogel [3] was used and the result was verified by studying the Apache Subversion [1] project. To assess the development practices within the Case Company the first researcher spent two months in the company, studying the company's processes and examining online communication trails and documentation.

Thus, a prolonged involvement [15] was applied in order to improve the validity of the research. The results of the documentation study were discussed and validated with the Case Company senior employees through a set of six structured interviews, i.e. member checking. The results were also reviewed by the second researcher who did not spend time in the company, i.e., peer debriefing. This also reduces the possible bias that the first researcher might have developed with a prolonged involvement. It also means that research triangulation was applied which also increases validity of the research. The participants included in Phase II of the study were the same individuals as the ones that participated in Phase I of the study, and thus were well acquainted with the subject of research and previous work

completed. The participants formulated the answers to the questions themselves, and participated in the discussion that followed which gave the opportunity to discuss and clarify their written answers even further. There exists possibility that participants were not representative sample, but the chances for this are very small.

The *internal validity* is concerned with causal relations. Since the nature of this study is to compare and analyze development practices, the causal relations are not seen as a threat of the study.

The *external validity* is related to the ability to generalize the results of the this study. The OSDP as defined in the paper can be relevant for future analysis. The Case Company studied is large software and hardware company, a world leader in its field, where the main products are built around OSS products. Hence, the participants are experienced in working with mature OSS communities. The competition in the market is typical. Hence, the findings of this research might be relevant to other large software companies that consider implementation of OSDP internally. It provides a framework of characteristics present in OSDP and an insight on benefits and challenges on implementing OSDP within a company setting.

The *reliability* aspect of validity is concerned with the aspect of data and analysis dependence of the underlying research on the researchers. The study was conducted as a prolonged, two phase, structured case study, with the analysis, interviews, and focus group meeting conducted in a structured way.

4. Results

4.1. RQ1

Research Question RQ1 concerns the changes in alignment of OSDP and software development practices of the Case Company over the past two years. To answer RQ1, the results of the first phase and second phase of the study are compared. The results are grouped under the three aspects as outlined in Table 1.

4.1.1 Infrastructure

The web portal is well structured and contains documents with information on organization structure, administrative information, roles and responsibilities, information on development processes, methods, standards, past, and ongoing project related information, code repository, use-net groups, and training manuals. Development processes and the methodology are well defined, with a project management process which is best categorized as a set of sequential steps The coding standards are clearly spelled out in the

documentation. There exists ongoing project documentation mostly with information on project management plans, allocated resources, assigned tasks, and task completion.

In the first phase of the study only two interview study participants indicated that they use the portal in their daily work, the architect and the senior project manager. The two interviewees used it for the purpose of updating project management plans or technical documents. At the same time developers, code block architects, and technical leads indicated that they used the portal very little in their daily project related tasks. They also agreed that the documents on the portal were not well organized and that much of the documentation they were interested in was out of date.

In the second phase of the study all focus group participants indicated that they use the company portals several times on daily bases. The portal is used to get up-to-date information on current bugs, fixes, and releases, to prioritize backlogs, to review design and architecture, to engage in communication about software projects, to follow mailing lists for different groups and projects, and as development wiki.

4.1.2 Communication

The internal portal also hosts infrastructure necessary to carry out discussions on various topics and create searchable archives.

In the first phase of the study the majority of the interviewees agreed that a majority of inefficiencies and issues they encounter in their daily work are related to inadequate communication. Most thought that better communication would lead to more efficiency at work. They expressed that usage of electronic communication in a standardized form would be desirable, especially if it would create searchable archives which could later on be referenced for problem solving purposes, similarly to how they would search the internet to understand why programs produce certain error codes and how such issues could be resolved. On the other hand, the majority of the interviewees agreed that it is much more time-efficient and easy to “go and talk” to a person about a problem, recognizing that in this way no written trail on the problem would be left. While there exist non-standardized means to communicate electronically, some interviewees said that the majority of developers refrain from using it, partly due to past experience, where questions and issues brought up through electronic discussions were not addressed in a manner that would facilitate such discussion.

In the second phase of the study all of the participants indicated that they engage in online communication for the purposes of solving ongoing issues, discussing ongoing project work, and searching for information through communication logs, mailing lists, on daily bases. The partici-

pants also noted that the increase in online communication was necessitated by opening of a new, global distributed development sight. They pointed out that in the distributed development environment the “go and talk to a person” option was no longer viable and they also recognized the benefits of having the communication archives of the discussions available as they provided a searchable information trail. This has also encouraged different area knowledge experts to produce wiki documents on the most frequently asked questions, which reduced time inefficiencies in repeating the same information.

4.1.3 Management

In the first phase of the study it was found that the organizational structure and roles and responsibilities within the R&D resemble roles which can be found in OSS communities. Hence, besides developers, there are technical leads, code block maintainers, code block architects, and architects. The code block maintainers code block architects and can be seen as fulfilling the roles of module maintainers and release manager (e.g. [7]) in the open source community. The majority of developers were positive towards the idea of being able to select tasks they would work on from a pool of tasks in the similar manner as this is done in OSS communities. However, interviewees that were in manager positions indicated that this might not be feasible as much time would then need to be spent on managing conflicts for those developers that could not choose tasks or were assigned to less interesting tasks. All interviewees agreed that task deadlines are needed, but sometimes too tight deadlines tend to negatively affect quality, as there then exists a tendency to put in much functionality without properly testing it.

Five of the interviewees thought that the number of formal meetings held was excessive. They expressed that if more time was put in planning of the meeting and appropriate selection of the attendees, the meetings might be less frequent and more efficient. Interviewees at more advanced technical position believed that there was a tendency to involve them into projects too early or too late.

Code block architects and code block maintainers noted that in practice their roles overlap with the role of technical lead. Such overlapping roles on the project are conflicting, as technical lead is perceived to be more of a project driver, while code block architects and maintainers are considered to be expert of a product or a part of it with a sole role of making sure that underlying product development is in line with overall architecture.

In the second phase of the study there were no changes noted with respect to the development roles which continued to be aligned with roles observed in mature OSS communities. The development teams are specialized in a part

of the platform and do not have the opportunity to initiate changes independently without consulting the platform owner. However, in case the changes are small, e.g. not affecting common APIs, not conflicting with customer requirements, and development resources are available, teams can independently implement the changes. An increased usage of online communication channels has positively affected the management aspect resulting in less unnecessary meetings, and greater information dissemination resulting in involving appropriate technical resources in early stages of project planning. However, the communication is still restricted to development resources, with platform owners serving as links to other project stakeholders or end customers. Hence, there is no direct feedback loop between the development teams and end customer, as is the case in OSS communities.

4.2. RQ2

Research question RQ2 concerns the underlying reasons for the changes in the OSDP implementation levels. Based on the results of the focus group meeting, the underlying reasons for change in level of the OSDP implementation practices is opening of a new distributed development site. This necessitated greater level of online communication has taken up the same characteristics as the ones observed in OSS communities. The increase in online communication manifested itself in greater usage of mailing lists, project wiki pages, and other online resources, e.g. for project management, bug reports, etc., thus transferring some of the work from 'live' interactions to an online milieu. This results in the creation of searchable archives, which further increased the development efficiency.

5. Discussion

The level of the alignment of the Case Company development practices with OSDP has increased over a two year period, with the change being driven by the need to achieve greater development efficiency across distributed development sites.

Basing its core software and hardware products on OSS and with development teams highly experienced and versed in OSDP, the Case Company has mimicked characteristics of OSDP as presented in Table 1. The company has replicated many of the roles present in OSS communities, such as the role of code block architects, technical leads, and head architects, and they have implemented an online portal and setup usenet groups. Development resources are aligned with OSDP in respect to the common understanding of technical issues and value of standardized practices in design, coding, testing, and development stages. Hence, in both stages of the study, experience gained from through

participation in the OSS community process were transferred back into the company.

Due to the lack of planned effort to encourage company-wide usage of an online communication milieu, in the first phase of the project it was observed that such resources are used scarcely. While a majority of developers preferred having searchable communication archives, a starting place where one could go to find out if there exists more information about an investigated issue, they perceive that communicating electronically instead of "face-to-face" is less efficient. The interviewees also indicated a reluctance to take part in open electronic discussions and such discussion are not formally encouraged or enforced to any degree. Face-to-face communication reduces the amount of time one needs to spend searching through archives and can also ask "on-demand" for further clarification of an issue. On the other hand, 'face-to-face' communication can be less efficient in case the resources one needs to talk to are not currently available. The second phase of the study showed significantly improved usage of company portals since they enabled more efficient communication and development between the existing and a new globally distributed sites. Transferring some of the 'face-to-face' communication to an online milieu created a searchable communication archives, and improved the documentation process, e.g. creating more documentation on the system and "F.A.Q. lists". The more transparent, online, nature of discussions on ongoing projects helped involve relevant resources in early stages of project planning, thus reducing the number of unnecessary meetings and ensured that relevant inputs are acquired early on.

The greatest misalignment still is evident in the way work is assigned and projects are managed. There is still a closed feedback loop between developers and outside company partners and customers, and much of this communication is channelled through higher, management level roles, such as platform owners. Developers only make up-stream change request, without involving platform owners in cases where the changes are minor and not affecting common APIs, provided that there resources available.

In the studied case, it seems like the matter of adopting OSDP is highly motivated and carried out by technical personnel that has recognized its benefits through experience with OSS communities. The management structures need to be further educated on the OSDP so an appropriate adoption process can be found and implemented, as this was the case presented in earlier studies, e.g. at HP [11] and Lucent [4].

6. Conclusions

The results of the prolonged case study show that as a company expends and acquires geographically distributed

development sites, adoption of OSDP is preferred and a natural way to engage all software product stakeholders in the most efficient way. This is a relevant finding, as with current need for highly skilled work force, many companies struggle with hiring appropriate human resources in one location, and are forced to either open new sites or outsource some part of development.

Unlike the previous studies which show how systematically planned OSDP implementation is carried out, we show what can happen when such process is carried out in a less planned way, driven by individuals in highly technical roles with extensive experience in working under the OSS process. Characteristics of such approach have shown that OSDP are implemented to a higher degree in a form of infrastructure, and less in a form of communication and management practices. Hence, there exist technical roles modeled around the software product, such as head architect and code block maintainer. There also exist standardized development practices and processes facilitating cross project work. The Case Company portal is created with the purpose of resembling an OSS community online milieu, but in practice, during the first phase of the study, the content of the portal was not well organized, complete, or up-to-date. However, with a new, geographically distributed development site, the second phase of the study observed that the online resources were used more, and thus more resembling OSDP. The greater transparency brought through increased online communication and work has also ensured that appropriate resources are involved in earlier stages of project planning, thus also reducing unnecessary meetings. Repetitive, time consuming tasks, such as 'go-and-talk-to expert' were reduced by having online resources, such as up-to-date documentation and a project wiki.

The prolonged case study presented in this research shows benefits and challenges of implementing OSDP within a closed company setting, when the adoption is not carried out in a systematic and planned way, but rather driven by employees in highly technical roles with experience of working under the OSS communities. We believe that there are more case companies undergoing the same challenges, especially as the software industry increases usage of OSS products and related business models. For this reason, more studies of similar type are needed, not only to raise awareness to the possible problems, but primary to better plan the adoption process so its full benefits can be taken advantage of.

References

- [1] ApacheSubversion. Apache subversion open source project. <http://subversion.apache.org/>, 2012.
- [2] B. Fitzgerald. The transformation of open source software. *MIS Quarterly*, 30(3):587–598, 2006.
- [3] K. Fogel. *Producing Open Source Software: How to Run a Successful Free Software Project*. O'Reilly Media, first edition, Feb. 2013.
- [4] V. K. Gurbani, A. Garvert, and J. D. Herbsleb. A case study of a corporate open source development model. In *Proceedings of the 28th International Conference on Software Engineering*, ICSE '06, pages 472–481. ACM, 2006.
- [5] V. K. Gurbani, A. Garvert, and J. D. Herbsleb. Managing a corporate open source software asset. *Commun. ACM*, 53(2):155–159, 2010.
- [6] M. Höst and A. Oručević-Alagić. A systematic review of research on open source software in commercial software product development. *Information & Software Technology*, 53(6):616–624, 2011.
- [7] C. Jensen and W. Scacchi. Role migration and advancement processes in ossd projects: A comparative case study. In *29th International Conference on Software Engineering (ICSE'07)*, pages 364–374, May 2007.
- [8] J. Kontio, J. Bragge, and L. Lehtola. The focus group method as an empirical tool in software engineering. In F. Shull, J. Singer, and D. I. K. Sjøberg, editors, *Guide to Advanced Empirical Software Engineering*. Springer, 2008.
- [9] I. Kuali Foundation. Open library environment. <http://www.kuali.org/ole>, 2016.
- [10] J. Lindman, M. Rossi, and P. Marttiin. Applying open source development practices inside a company. *IFIP International Federation for Information Processing*, 275:381–387, 2008.
- [11] C. Melian and M. Mähring. Lost and gained in translation: Adoption of open source software development at hewlett-packard. In B. Russo, E. Damiani, S. Hissam, B. Lundell, and G. Succi, editors, *IFIP International Federation for Information Processing*, volume 275, pages 93–104. Springer, 2008.
- [12] A. Oručević-Alagić and M. Höst. A case study of open source development practices within a large company setting. *ICSET 2014 International Conference on Software Engineering and Technology, Istanbul, Turkey, Sep 29-30, 2014*, 2014.
- [13] E. S. Raymond. *The Cathedral and the Bazaar*. O'Reilly Media, Inc., 2001.
- [14] C. Robson. *Real World Reserach*. Blackwell Publishing, 2:nd edition, 2002.
- [15] P. Runeson, M. Höst, A. Rainer, and B. Regnell. *Case Study Research in Software Engineering*. Wiley, 2011.
- [16] W. Scacchi. The future of research in free/open source software development. In *FoSER*, pages 315–320, 2010.
- [17] K.-J. Stol, M. A. Babar, P. Avgeriou, and B. Fitzgerald. A comparative study of challenges in integrating open source software and inner source software. *Information & Software Technology*, 53(12):1319–1336, 2011.
- [18] K.-J. Stol and B. Fitzgerald. Inner source–adopting open source development practices in organizations: A tutorial. *IEEE Software*, 32(4):60–67, July 2015.
- [19] The Apache Software Foundation. Hadoop. <http://hadoop.apache.org>, 2016.

A Novel Open Source Software Ecosystem: From a Graphic Point of View and Its Application

Chenxi Song, Tao Wang, Gang Yin, Xunhui Zhang, Cheng Yang
National Laboratory for Parallel and Distributed Processing
College of Computer, National University of Defense Technology
Changsha, China

songchenxi92@163.com, { taowang2005, yingang }@nudt.edu.cn, zhangxunhui9368@163.com, delpiero710@126.com

Abstract—With the rapid development of open source software, various elements such as OSS, developers, users and online posts, across different communities and their interactions constitute a novel software ecosystem. Most of the current researches about software ecosystems care the connections between software, and few of them consider the relationship across communities from an overall perspective, and fail to cover the users and their activities which should be an indispensable part of the OSS ecosystem. This paper model the OSS ecosystem as a graph, which combines different types of OSS communities as a whole. Based on this graph model, we analyze the characteristics of ecosystem, like the evolution, competition and symbiosis. In addition, we build a recommendation system as well, and the experiment results suggest the validation of our approach.

Keyword; *open source ecosystem; graph model; recommendation algorithm*

I. INTRODUCTION

In recent years, there has been a surge of interest in open source software (OSS, for brief) development and most of them focus on single community. As far as we know, there are a large number of communities related to open source software, such as GitHub, StackOverflow, OpenHub, OSChina, etc. Some of them focus on project developing and software hosting, like Github, and others focus on knowledge sharing, like StackOverflow. Therefore, we divide these communities into two categories, software development community that can provide abundant repository data and development process reports, and knowledge sharing community that contains a large amount of discussion about OSS. In this paper, we connect different communities together, constitute an OSS ecosystem, extract various types of OSS related data and organize them into a graph model.

Since Messerschmitt and Szyperski in [1] propose the conception of *software ecosystem* in 2003, the researches on software ecosystems have been around for more than a decade. Surveys [2] [3] about different definitions and research emphasis of software ecosystem show that there has no consistent definition through more than 20 research works. These works all contain a collection of software and some kinds of relationships either symbiosis, dependency [5], common evolution [4], business or technical. To the best of our

knowledge, few of them combine different communities together taking advantage of the feedback from users.

In this paper, we propose a novel definition of open source software ecosystem, which combines different elements from a variety of open source communities for the first time. Firstly, we construct a graph model to describe the software ecosystem, in which each element is seen as a vertex, relation as edge. Secondly, we analyze characteristics of the ecosystem such as the symbiosis, competition between software. Afterwards, from the graphic view, we build a recommendation system that recommends related software to users. Compared with previous work, our ecosystem covers a variety of open source communities. Combining different communities together, we can analyze the development process of OSS and relations between them like cooperation and competition, through a global perspective.

This paper is organized as follows: section II describes the definition and graph model of OSS ecosystem as well as the recommendation approach; section III introduces the experiments of analyzing characteristics of the ecosystem; experiment results and the evaluation of recommendation are in section IV, followed by a discussion of threats to validity; we conclude with a discussion of our contribution in section V.

II. OSS ECOSYSTEM

This section will introduce the definition of our open source software ecosystem. Since different elements in the ecosystem interconnect with each other, we map the ecosystem to a graph model. Based on the graph, we design a recommendation system, which is described in this section as well.

A. Background and Definition

In ecology, an ecosystem is a community of living organisms in conjunction with the nonliving components of their environment (things like air, water and mineral soil), interacting as a system [10]. In open source area, there exists elements of software development community like software, developers and elements of knowledge sharing community like online posts, users, followers etc. The associations between elements constitute an OSS ecosystem.

We define the OSS ecosystem as **a set of open source software along with their contributors, followers, users, and the social message generated by the development activities**

and users feedback. The ecosystem runs as Fig.1, in which the direction of arrows expresses the flow of products.

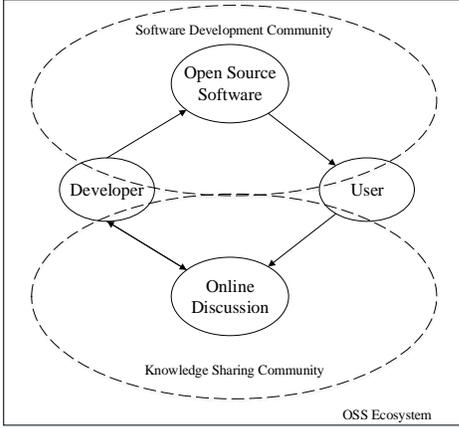


Figure 1. Actors in the OSS ecosystem

In ecology, ecosystem contains producers, consumers and the ecosystem dynamics, which are presented in the interactions between different species [4]. Besides the food chain of natural ecosystem, there also exist relationships like symbiosis, competition, etc. In our software ecosystem, these characters and relationships have corresponding expressions.

The producers of our software ecosystem are, of course, the developers and the software itself. Consumers are users. However, unlike the consumers in natural ecosystem, the software consumers also give feedback to producers, share their use experience, report bugs and make demands through online posts and other messages. The food chain is the flow of OSS. Symbiosis and competition mainly exists between OSSs, and we will introduce them in section IV.

B. Graph Model for OSS Ecosystem

The ecosystem contains developers, OSS, users and online posts in general. Since tag is very common in communities, we also take it into consideration. Based on the interactions between different elements, we map the ecosystem to a graph model, represented as $G = (V, E)$. In the directed graph G , each vertex $u \in V$ represents an element of open source ecosystem and the set of edges E contains all relationships that are present between these two distinct sets of vertexes. Both vertexes and edges can have labels and properties. The graph model is shown in Fig. 2.

Five types of vertexes exist in the graph model, namely: *Software*, *Tag*, *Community*, *Post* and *Person*. *Software* is the essential element, *Tag* is label of software or posts in communities for category or subject indexing, and *Community* identifies the source of software or posts.

The seven kinds of edges are based on relationships and interactions between elements. The direction and property of edges can be seen from Fig.2. Edges are the important bases for the analyzing of OSS ecosystem. For instance, the *DISCUSS_ABOUT* edge is defined as

$$e_{DISCUSS_ABOUT} = \{ \langle u, v \rangle \mid u \in V_{Post}, v \in V_{Software} \},$$

which means post u is a discussion of software v . Particularly worth mentioning is that we have a specific algorithm to bridge open source mentioning and online posts.

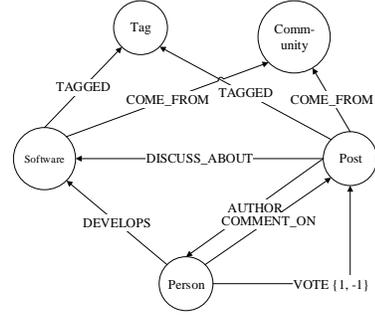


Figure 2. The graph model of OSS ecosystem

C. Recommendation Approaches

A primary application of the OSS ecosystem is a recommendation system that can recommend relevant software for a specific software that user required. The general approach is that once a user require for a specific software s_0 , we first calculate the relevancy between s_0 and other software, and recommend the most relevant ones.

The relevancy between two OSSs is calculated based on two aspects, namely: tags and posts, denoted as r_t, r_p . Fig.3 is a diagram of the two relevancies.

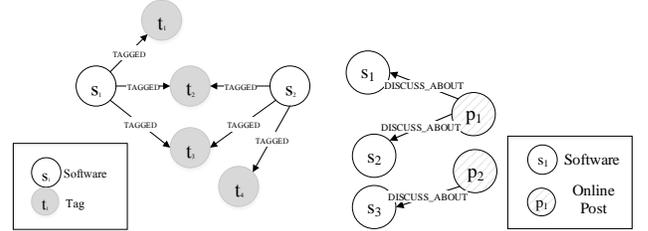


Figure 3. Relevancy calculation

Tag relevancy: After analyzing the tag usage of OSS ecosystem, we find that tags indicate the technology, language or feature of the software. So containing same tags shows some relevance of two OSSs. However, some tags like “java” is so common that only calculating the intersection may lead to incorrect OSSs. So we draw lessons from TF-IDF and calculate a word frequency weight for each tag.

For the tag t_i of OSS s_j ,

$$tf_{ij} = \frac{n_{ij}}{\sum_k n_{kj}} = \frac{1}{\sum_k n_{kj}} \quad (1)$$

$$idf_{ij} = \log \frac{|S|}{|\{j:t_i \in s_j\}|} \quad (2)$$

$$w_{ij} = tf_{ij} \times idf_{ij} \quad (3)$$

In TF-IDF model, n_{ij} of Eq. (1) means the number of appearances of tag t_i in software s_j , which is 1 in our scenario. And $\sum_k n_{kj}$ represents the total tag number of s_j . In Eq. (2), $|S|$

is the total number of OSSs, and $|\{j: t_i \in s_j\}|$ represents the number of OSSs that contain tag t_i . The TF-IDF value is used as the weight of each tag and transforms the set of tags into vectors. Then *cosine similarity* is used to calculate the tag relevancy between two software.

Post relevancy: As we mentioned before, users may post in communities to discuss one or more open source software, which is indicated by the *DISCUSS_ABOUT* relationship in the graph model. OSSs with related or similar function are often discussed together, we call this phenomenon the co-occurrence of software. After analyzing the different OSSs discussed in the same post, we find that these OSSs often have related functions or same user groups, the statistics is shown in section V. So we treat the co-occurrence phenomenon as an important basis of recommendation.

For a particular software, we first gather all OSSs that are discussed with s_0 in same posts and their co-occurrence frequency. The relevancy of posts defines as the normalized frequency, whose range is between 0 and 1.

Recommendation Model: In order to synthetically consider the influence of two factors, we assign each relevancy a weight value, α and β . And calculate the final recommendation score as:

$$R = \alpha * r_t + \beta * r_p \quad (4)$$

Find the top-k list of the final relevancy score R with regard to the specific software as recommendation result.

III. EXPERIMENTS DESIGN

Symbiosis, competition and evolution are significant features of ecosystem in ecology, which have corresponding expressions in our OSS ecosystem. In this section, we present our research questions and experiment settings of analyzing ecosystem characteristics.

A. Research Questions

OSS ecosystem connects different communities together, which gives us an opportunity to analyze the characteristics of open source environment in multiple views. The bridge between open source software and online posts remarkably leverage the crowd wisdom to support the development of OSS. In this section, we would like to analyze the OSS ecosystem in three aspects.

1) The symbiosis and competition of open source software

In software ecosystem, different software with same or similar functions may have competitive relationships, like Ubuntu and Fedora. Collaborations exist as well, such as MySQL and the MySQLWorkbench, which is called symbiosis in natural ecosystem. We located the two relationships in software ecosystem through co-occurrence phenomenon.

2) The evolution of open source software

Species in ecosystem evolve over time to adapt to the changing environment and reproduce, so does open source software in our ecosystem. Discussions from users is an important reflection of OSS evolution. We study the change of discussion number and discuss the evolution of OSS.

3) A recommendation application based on graph model

Based on the graph model and characteristics mentioned before, we accomplished a recommendation application to propose related software when users require for a specific OSS.

B. Dataset and Experiment Settings

In order to collect enough OSS information for establishing an integrated OSS ecosystem, we use a web crawler to gather realistic sets of OSS data from dozens of communities and get a total of more than 230,000 software and 5,100,000 online posts. Since the data quality in these websites varies greatly, we select the valuable and filter out the noises. At last we obtain about 11,000 software and 100,000 online posts and its users, tags as well as information of communities for building the OSS ecosystem and conducting the experiments.

IV. EXPERIMENT RESULTS

Our ecosystem associates online posts with open source software, which gives us a chance to seek the overview of the ecosystem. In this section, we analyze the characteristics in OSS ecosystem and evaluate the recommendation system. Meanwhile we describe some threats to validity.

A. Ecosystem Characteristics

Limited by the data source, we cannot build the relations between OSSs (Fig.2). However, we can still catch a glimpse of the relationships of OSS with the aid of discussion posts.

1) The symbiosis and competitive relationships

When online posts discussing two or more software at the same time, these software must have some connections. For example, app developers may share their experience in using *SQLite* while talking about the feature of *Android*. These two software cooperate in developing Android apps. In this occasion, we think the co-occurrence software have symbiosis relationship. Also, users may compare several similar software, and these software are competitors.

We collect 500 sets of software that has co-occurrence relationship and find that 87.2% of them are the symbiosis relationship. Only 9% are competitors and others 3.8%.

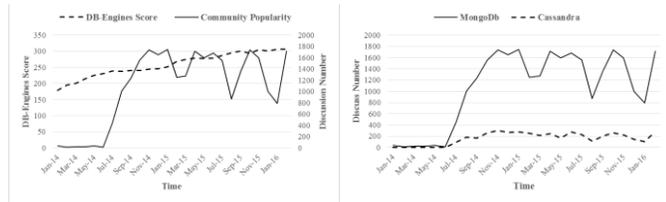
2) The evolution of open source software

Darwin, the famous naturalist and geologist, proposed “Survival of the fittest” evolution theory. It also fit for software ecosystem. Similar to the method in 1), online posts are the basis element of analyzing software evolution.

Firstly, a popular software may be discussed more than others. Take *MongoDB*, a popular NoSQL database, as an example. We gather statistics of its discussion number of each month and compare it with the scores given by DBEngines¹, a famous DBMS ranking website (Fig. 4(a)). The fluctuation of discussion number may be caused by some events, for example, a new version released. An apparent ascends in April 2015 may due to the release of 3.0. Each time a new version released users enjoy discussing the new features. As time goes by, this enthusiasm of the discussion may gradually reduce. Hence, the discussion number can represent the development of software to some extent.

¹ http://db-engines.com/en/ranking_trend/system/MongoDB/

Also, the evolution of software may influence the development of competitors. Fig.4 (b) shows the growing trend of *MongoDB* and *Cassandra*. As competitors, *MongoDB* gets the support of more users and has higher rank in DBEngines.



(a) The community popularity and DB-Engines score of mongoDB (b) The comparison of discussion number

Figure 4. The evolution of software

B. Recommendation Evaluation

In order to evaluate the performance of recommendation system objectively, we compare our output with the recommendation of OSChina. OSChina² is the biggest open source community of China, which is one of our data sources. Take *MySQL* as an input example, the recommendation results of two algorithms is shown in Table I.

TABLE I. MYSQL RECOMMENDATION RESULTS

Our approach	OSChina
- PDO	- MySQL MTOP
- phpMyAdmin	- MySQL Installer
- Hibernate	- Google MySQL
- WordPress	- MySQL Syncer
- MySQL Connector/J	- MySQL Utilities

As Table I shows, OSChina recommend MySQL tools which used to improve the performance in data storage, management and synchronization. However, it seems that these approaches pay too much attention to the software itself. Our approach, by contrast, provides a more practical choice. *PDO* and *PhpMyAdmin* are widely used for accessing and managing MySQL dataset in *PHP* software developing. *Hibernate* and *MySQL Connector/J* are the encapsulation for *JDBC* in *Java* applications development. *WordPress* is web software to create blogs and apps, which use *MySQL* to do data storage.

The options that our recommendation system give, are popular developing tools cooperating with *MySQL*. When a user searches for *MySQL*, he may need these interfaces or management tools to set up the whole system. The advantage of this result is that when a user requires a specific software, some popular and related tools will be recommended. It helps users to notice the hot technologies of related software and there is high possibility that they are using or intend to use one of them.

C. Threats to Validity

Firstly, the data in knowledge sharing communities may have some noise. The online posts or technique news don't have to be only related to open source software. It may also talk about some latest technology or some business software. We have designed a particular program to filter these data and

minimize the noise. Secondly, the relationship between open source software and online post is established by a match algorithm which conduct textual analyze to find the software that a post discuss about. In our testing, the bridging algorithm has a high accuracy of about 90%, and we filter some error before using to minimize the impact of errors. But it can still bring adverse effect to the ecosystem and its analysis. Also due to our limited datasets and parameters used in our approach, the evaluation is not comprehensive enough.

V. CONCLUSION

Software ecosystem has been a hotspot in software engineering area. In this paper, we collect millions of open source software, online posts and user information from dozens of open source communities. The association and interaction from the variety of data constitute an OSS ecosystem. We study the software evolution, as well as the symbiosis and competition between software, which are the important notions in ecology ecosystem. Viewing the elements in the ecosystem as vertexes, the relations as edges, we map the software ecosystem to a graph model, which helps us to get a full appreciation of the distribution and characteristics of ecosystem. Also, based on the analysis and the graphic view, we propose a novel recommendation approach and evaluate the result critically.

ACKNOWLEDGMENT

This research is supported by the National Natural Science Foundation of China (Grant No.61432020, 61472430 and 61502512).

REFERENCES

- [1] D. Messerschmitt and C. Szyperski, *Software Ecosystem: Understanding an Indispensable Technology and Industry*. MIT Press, 2003.
- [2] K. Manikas and K. Hansen, "Software ecosystems – A systematic literature review", *Journal of Systems and Software*, vol. 86, no. 5, pp. 1294-1306, 2013.
- [3] A. Serebrenik and T. Mens, "Challenges in Software Ecosystems Research", in *European Conference on Software Architecture*, 2015, p. 40.
- [4] T. Mens, A. Serebrenik and A. Cleve, *Evolving Software Systems*.
- [5] L. Šubelj and M. Bajec, "Community structure of complex software systems: Analysis and applications", *Physica A: Statistical Mechanics and its Applications*, vol. 390, no. 16, pp. 2968-2975, 2011.
- [6] G. Yin, T. Wang and H. Wang, "OSSEAN: Mining Crowd Wisdom in Open Source Communities", in *Service-Oriented System Engineering*, 2015, pp. 367-371.
- [7] S. Bajracharya, J. Ossher and C. Lopes, "Sourcerer: An internet-scale software repository", in *ICSE Workshop on Search-Driven Development-Users, Infrastructure, Tools and Evaluation*, 2009, pp. 1-4.
- [8] H. Wang, T. Wang, G. Yin and C. Yang, "Linking Issue Tracker with Q&A Sites for Knowledge Sharing across Communities", *IEEE Transactions on Services Computing*, pp. 1-1, 2015.
- [9] Y. Yu, H. Wang, G. Yin and T. Wang, "Reviewer recommendation for pull-requests in GitHub: What can we learn from code review and bug assignment?", *Information and Software Technology*, vol. 74, pp. 204-218, 2016.
- [10] "Ecosystem", Wikipedia. [Online]. Available: <https://en.wikipedia.org/wiki/Ecosystem>.

² <http://www.oschina.net/>

An FCM-based Personalized Affective Model for Agile Software Development

Wenjing He^{1,2}, Jun Lin^{3,4}, Xinjia Yu³, Zhiqi Shen³, Chunyan Miao³

¹Institute of Computing Technology, Chinese Academy of Sciences, China

²University of Chinese Academy of Sciences, China

³Joint NTU-UBC Research Centre of Excellence in Active Living for the Elderly (LILY)

Nanyang Technological University, Singapore

⁴College of Software, Beihang University, Beijing, China

hewenjing@ict.ac.cn, {junlin, xyu009, zqshen, ascymiao}@ntu.edu.sg

Abstract—Developers’ emotional stability in agile software development (ASD) teams is an important factor affecting the success of a project. Traditional Happiness Chart depends on people’s self-report and is unable to produce predictive analysis to facilitate decision-making. In this paper, we proposed a Fuzzy Cognitive Map (FCM) based method for analyzing ASD developers’ emotional stability. We designed an easy-to-use survey questionnaire which can help establish the values of the causal relationship weights to support personalization of the proposed FCM model. The FCM affective model produces two metrics for assessing a given user’s emotional stability: 1) the stable state mood value, and 2) the number of iteration steps for the FCM model to reach equilibrium. The two metrics are evaluated through a real-world experiment involving 20 undergraduate students. The results show that they are strong explanation factors for a user’s emotional stability with 99% confidence. A regression model based on the user study has also been presented in the paper to help researchers study the relationships between the two metrics and a person’s emotional stability computationally in the future.

I. INTRODUCTION

Human factors are important to the success of agile software development (ASD) [1], [2]. ASD team members, especially programmers, are not very keen on expressing their emotions. Nevertheless, team members have emotions. During the development process, their emotional states can be affected by many factors, such as work expectation, delays, quality of software artifacts, etc. The changes in emotion, in return, further affect the software development processes and the results.

In real-world projects, practitioners are starting to take note of this and attempt to use the Happiness Chart (<http://agiletrail.com/2011/09/12/how-to-track-the-teams-mood-with-a-niko-niko-calendar/>) to monitor team members’ daily mood and encourage emotional openness within the team. An obvious issue with this approach is that it is difficult to know whether one developer’s mood will be influenced by the emotions written down by another developer. Other important aspects include accuracy, authenticity, and congruence etc. are also not considered in the method.

In this paper, we investigate the impact of ASD team members’ mood stability which has been found to be important for learning, task performance [3] and building trust

[4], [5], [6]. We propose a Fuzzy Cognitive Map (FCM) based method of mood analysis. To enable personalization of the proposed FCM model, we also propose an easy-to-use survey questionnaire which can help establish the values of the FCM causal relationship weights. The FCM affective model is capable of computing two metrics for a given user: 1) the stable state mood value in the range of $[0, 1]$ (with 0 indicating very low mood and 1 indicating very high mood) and 2) the number of iterations for the FCM model to reach equilibrium.

To evaluate the explaining power of the two metrics of the proposed FCM affective model, we conducted the a real-world experiment involving 20 undergraduate students formed into ASD teams in Beihang university, China. The results show that the two metrics are strong explanation factors for a user’s emotional stability with 99% confidence. A regression model based on the user study has also been presented in the paper to help researchers study the relationships between the two metrics and a person’s emotional stability computationally. The proposed method is a promising approach for analyzing ASD team members’ moods.

II. RELATED WORK

Emotions in human communication process are very important for perception, decision-making, interaction, and intelligence. Modelling and analyzing ASD team members’ mood is essential to reducing the impact of developers’ mood swings on the development process. However, there is few research work currently focused on this direction. In this section, we first discuss the general research field of affective computing, and then focus on affective modelling in ASD.

A. Affective Computation Models

The Ortony, Clore and Collins (OCC) model [7] is the most widely adopted affective model. It regards emotions as the results of the following three types of subjective appraisals:

- 1) *The pleasantness of the consequences of events with respect to the agent’s goals*: emotions that we call goal-based or event-driven emotions will be stimulated by valenced reactions to consequences of events and appraised by agents’ internal goals, including “happy-for”, “resentment”, “gloating pity”, “hope”, “fear”, “satisfaction”,

“fears-confirmed”, “relief”, “disappointment”, “joy” and “distress”;

- 2) *The approval of the actions by the agent itself or another agent with respect to a set of behavior standards*: emotions that we call standard-based or agent-driven emotions will be stimulated by valenced reactions to actions of agents and appraised by agents’ internal standards, including “pride”, “shame”, “admiration” and “reproach”;
- 3) *The liking of objects with respect to the attitudes of the agent*: emotions that we call attitude-based or object-driven emotions will be stimulated by valenced reactions to aspects of objects and appraised by agent’s internal attitudes, including “love” and “hate”.

In general, emotions are affected by both internal and external factors. Internal factors include goals, preferences and attitudes, whereas external factors include events, conditions and other objects in the environment.

Based on the OCC theory, a probabilistic model that assesses student emotions during a game was proposed in [8]. The model can predict a player’s emotional state by assessing the player’s appraisal of interactions with the game, in light of the player’s goals and personality. In [9], an emotional agent was proposed for a serious game based on the Goal Net model [10]. The agent’s emotions are modeled by the OCC model powered by an FCM inference engine. Composite emotions from players of a game have also been studied in [11], [12], [13], [14], [15], [16]. The results provide researchers with regression models to infer player emotions based on their observed behaviour in a competitive game environment.

B. Emotion Models in Agile Software Engineering

An ASD team consists of different roles and people’s emotions may significantly affect the outcome of their activities, including those activities in the software development process. The team leader knows that the emotional status of members will affect their tasks and artifacts, but human’s emotions and their transition are too complex to observe and control [17].

In [18], the authors attempted to use the affect grid psychological tool to characterize emotions in software requirement engineering. Their results revealed that emotions were key issues in software development activities, and the importance of emotion management in software development life-cycle is important as software development is a human activity. Knowing the emotional state of the development team helps the project manager to create a conducive environment or implement suitable incentives for the team to avoid the effects of undesirable emotions. However, there is current a lack of computational affective models suitable for use in ASD.

III. PRELIMINARIES

According to the OCC theory, we know that a developers’ mood or emotion is affected by specific events, conditions or other situations in the ASD process. Therefore, we want to build an affective model for ASD team members to simulate interactions between development activities and developers’

moods. During our survey, we found modelling and computational capabilities of Fuzzy Cognitive Maps (FCMs) are suitable for building a causal relationship model for the ASD process. This may provide the agile team the ability of viewing emotional state transitions and changes during the process. In this section, we will first introduce the basics of FCMs.

An FCM is a fuzzy-graph structure for causal reasoning. It was developed from the concept of a Cognitive Map (CM) [19]. CMs were used to model a system with concepts and cause-effect relationships. The relationships can be divided into three types: 1) *positive*, 2) *negative*, or 3) *neutral*. CMs can be drawn as a directed graph, the nodes corresponding to relevant concepts/variables in the given domain, and the directed edges denote the relationships between two concepts/variables. The type of a relationship is denoted by a sign associated with the edge. A positive sign indicates a positive relationship with a *promoting effect*. This means that an increase in the value of the start concept value will lead to an increase in the value of the end concept. Similarly, a negative sign means a negative type with the *inhibiting effect*. This means an increase in the value of the start concept leads to a decrease in the value of the end concept. No connection between two concepts means that the two concepts are independent from each other.

CMs have two main drawbacks: 1) they do not allow feedbacks; 2) the relationships are binary. Therefore, the use of cognitive maps to model complex systems is infeasible. To tackle them, the Fuzzy Cognitive Maps (FCMs) were proposed as an extension. Compared to CMs, FCMs have two significant enhancements:

- 1) Causal relationships between concept nodes in an FCM are fuzzified. This character enriches the description of the links by numerical value instead of only using positive, negative and no connection. It allows various degrees to denote different causal influences.
- 2) FCMs are models for expressing dynamic systems. FCMs can evolve with time, and allows feedback mechanisms. Specifically, the effect of changing the value of one concept node in the model may change the values of other concept nodes, and the change could loop back to the original node in subsequent time steps.

The strength of relationship between two FCM nodes takes on any value in the $[-1, 1]$ range. A value of -1 represents the fully negative causal effect, whereas $+1$ means a fully positive causal effect. Zero denotes no causal effect. Other values denote different fuzzy levels of causal effect. The knowledge of system relationships can be described by a matrix, called the connection matrix. Each cell of this matrix stores a value of a corresponding relationship. The commonly used convention is to place the start nodes in rows and the end nodes in columns.

For example, considering a system with N concept nodes, we have an $N \times N$ matrix representing the FCM. The elements in the matrix are the strengths of the causal relationships. Any one state of the system can be determined by one state vector, which specifies the current values of all system concept nodes. The FCM iteratively updates the state of the system. In one

iteration, the value of each node is calculated based on the current values of every node exerting influence on it through its causal relationship. After multiplying these values by the edge weight between the two nodes, the sum of these products is taken as the input to a transformation function, which is used to reduce the unbounded inputs to a certain range.

The value of each node in any iteration is computed from values of nodes in the preceding state, using the following equation:

$$N_j(k+1) = f\left(\sum_{i=1}^n e_{ij}N_i(k)\right) \quad (1)$$

where $N_i(k)$ is the value of the i th node in the k th iteration; e_{ij} is the edge weight between nodes N_i and N_j ; n is the number of concepts in the FCM; and $f(\cdot)$ is the transformation function. Three types of transformation functions are commonly used. They are the binary, trivalent and sigmoid functions [20]:

1) *Bivalent* function

$$f(N_i(k)) = \begin{cases} 1 & , N_i(k) \leq 0 \\ 0 & , N_i(k) > 0 \end{cases} \quad (2)$$

2) *Trivalent* function

$$f(N_i(k)) = \begin{cases} -1 & , N_i(k) \leq -0.5 \\ 0 & , -0.5 < N_i(k) < 0.5 \\ 1 & , N_i(k) \geq 0.5 \end{cases} \quad (3)$$

3) *Sigmoid* function

$$f(N_i(k)) = \frac{1}{1 + e^{-cN_i(k)}} \quad (4)$$

The final results of a simulation performed with FCM strongly depend on the transformation function. If we use the function which results in binary values, the simulation of a FCM system leads to either fixed state pattern of node values (which is called hidden pattern or fixed-point attractor) or a cycling between several states (which is known as the limit cycle). If we use a continuous-output transformation function, the simulation may result in a different outcome. The system may continue to produce different state vector values for successive cycles. In this case, this unstable situation is called a chaotic attractor [21].

The FCM theory has been applied in many domains for more than two decades to model systems or environments with complex causal relationships. These domains include software project management [21], software quality risk analysis [22], software education [23] and agile software development [24].

IV. THE PROPOSED METHOD

A. Method

As FCM theory and method can be used to build causal relationship models for complex systems or processes, we will use it to build affective models for ASD teams, and then use their computational functionality to simulate the interactions between development activities and human moods.

According to OCC theory, external events or activities can affect developers' emotion or mood. Therefore, events (and

event results) and developers' mood will be incorporated into the proposed model. Our method includes the following steps:

- *Step 1:* The team manager designs a general emotional FCMs model for all team members. The concept nodes will include developers' mood, external events/activities and their results. There are two ways to determine the weights between concepts: 1) self-reports by members, and 2) based on members' past behaviour records.
- *Step 2:* The system executes the FCMs to reach the equilibrium state for each team member. The final state can be treated as a developer's personalized mood level, and the time to reach the state can be used to assess the developer's capability for mood control.
- *Step 3:* The system monitors the changes at the early stage (when the node values change rapidly), and how many iterations the model takes to reach the equilibrium state for each individual.
- *Step 4:* The system shows a comparison score list to help the team manager predict who will be the most suitable candidate to handle a given ASD activity.

To simplify the model, we consider their complex personal emotions as one comprehensive mood factor. According to common ASD experience and ignoring other factors, the causal relationships between developer's mood status, progress of task execution and quality of task execution during development process are shown in the FCMs in Figure 1(a). The general model presents the relationships that are essential during an ASD process, which consists of three concept nodes as follows:

- *C1:* Worker's Mood Status (called "Mood" for short) reflects the comprehensive feelings to an event or an event result. The mood value can include "high", "medium" and "low" or finer granularities. They affect or are affected by the results of task executions. The mood values can be acquired by the *AffectButton* [25] and then normalized in a range of $[0, 1]$.
- *C2:* Task Execution Progress (called "Progress" for short) can be interpreted as a relative comparison concept for the task execution progress. Its value can be calculated by the ratio of actual task completed time to estimated task completed time, and then normalized into a range of $[0, 1]$.
- *C3:* Task Execution Quality (called "Quality" for short) can be interpreted as a relative comparison concept for artifact quality after task execution. Its value can be acquired through a rating assessment by other team members, and then normalized into a range of $[0, 1]$.

The casual relationships between nodes are represented by directed edges with fuzzy weights based on human experience acquired through questionnaires.

B. Illustrative Examples

To illustrative the proposed method, we look into two fictitious developers, "Michael" and "Grace". From the FCM model (Figures 1(b) and 1(c)), we can see the mood of the

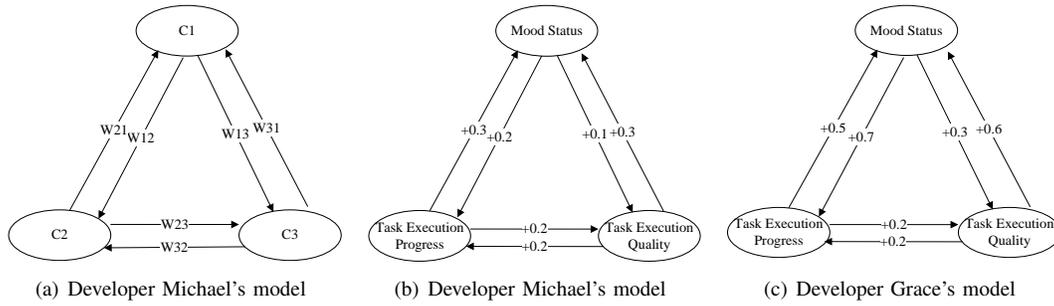


Fig. 1. The proposed general developers' FCM model.

developer who picks up the task has positive effect on the task execution progress (an edge weight of 0.2 from $C1$ to $C2$ for Michael; and an edge weight of 0.7 from $C1$ to $C2$ for Grace). This means the progress of task execution will be good when the developer's mood status is at a high level. Meanwhile, the task execution progress also positively influences developers' mood (an edge weight of 0.3 from $C2$ to $C1$ for Michael; and an edge weight of 0.5 from $C2$ to $C1$ for Grace). This means developers mood is low when the task is delayed and vice versa. This conforms to the common sense in real projects.

Furthermore, the developers' mood positively impacts the task execution quality (an edge weight of 0.1 from $C1$ to $C3$ for Michael; and an edge weight of 0.3 from $C1$ to $C3$ for Grace). On the other hand, the increase in quality can lead to higher mood among the developers (an edge weight of 0.3 from $C3$ to $C1$ for Michael; and an edge weight of 0.6 from $C3$ to $C1$ for Grace). Generally, good task execution progress also leads to high quality (an edge weight of 0.2 from $C2$ to $C3$ for Michael; and an edge weight of 0.2 from $C2$ to $C3$ for Grace). High quality software with fewer bugs will make the overall progress faster (an edge weight of 0.2 from $C3$ to $C2$ for Michael; and an edge weight of 0.2 from $C3$ to $C2$ for Grace). The strengths of the relationships were established based the developers' personal experience.

The FCM-based model for Michael can be presented in a matrix:

$$\begin{bmatrix} 0 & 0.2 & 0.1 \\ 0.3 & 0 & 0.2 \\ 0.3 & 0.2 & 0 \end{bmatrix}. \quad (5)$$

The matrix for Grace's FCM model:

$$\begin{bmatrix} 0 & 0.7 & 0.3 \\ 0.5 & 0 & 0.2 \\ 0.6 & 0.2 & 0 \end{bmatrix}. \quad (6)$$

Next, simulations are run based on the FCM models. The starting vector is denoted as Iteration #0. Each state vector consists of three numbers, which correspond to the conceptual nodes Mood ($C1$), Progress ($C2$), and Quality ($C3$). The simulation begins with the start state vector Iteration #0 = (0.5, 0, 0) for both developers, which represents a situation in which Mood is active and set at a medium value of 0.5, and other concepts are inactive (i.e. the developer has not yet started to work). As the simulation continues, successive values of the nodes show trends which occur with

the progressing time. By analyzing the states of the nodes in the simulation process, observations obtained and analyzed.

The simulation was carried out using the logistic sigmoid function as a threshold function, which is a continuous-output transformation function and thus provides true fuzzy conceptual node states. We mainly want to see the comparative results between two developers, so the constant c can be set to a common number 5.

For Michael, the model reaches equilibrium at Iteration #14 = (0.92058, 0.846519, 0.786979). For Grace, the model reaches equilibrium at Iteration #7 = (0.994717, 0.987922, 0.922728). Figure 2 shows the results of Michael's model in a plot, and Figure 3 shows the results of Grace's model in a plot.

This model describes state change trends in a task development activity. It characterizes a situation when a worker starts to do a task. Successive states of the modeled system show changes in concept node values, which represent workers' mood and task execution aspects. The final state achieved by the model shows concept nodes in equilibrium states. This can be considered as the prediction by the model on the likely outcomes of the development activities and the developers' mood.

The first iteration (#1) shows the beginning of execution of the ASD activities. As time goes by, the workers' mood and quality of work increase rapidly at first. At iteration #1, the workers need some time to familiarize with the new task and environment. Therefore, their progress was not fast and mood

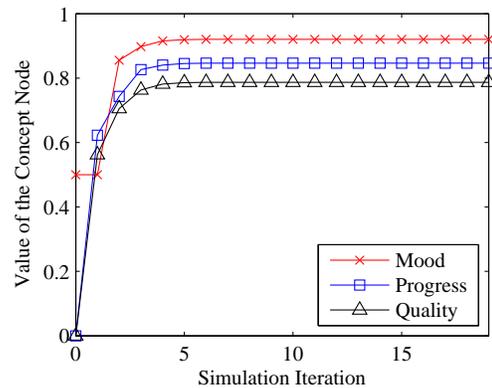


Fig. 2. Results of Michael's first model.

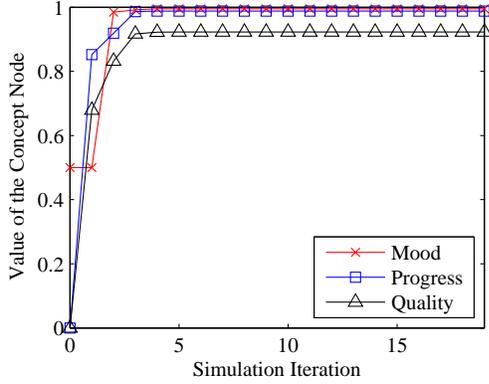


Fig. 3. Results of Grace's first model.

is not high. After this period, the mood value increases and then stays at a stable level, especially for Grace. For Michael, after the quality reaching the highest value at Iteration #12 and the progress reaching the highest value at Iteration #11, the mood reached the equilibrium value at iteration #14. For Grace, before the quality reaching the highest at Iteration #7, the mood and the progress both reach the highest values at Iteration #6 and then stay at the equilibrium. This result shows that the Grace needs less time to reach the stable state, and her final state performs better than Michael.

V. EMPIRICAL EVALUATION

We have engaged 20 undergraduate software engineering students taking the ASD course in our user study. Each student completed the questionnaire so that the proposed model can be personalized to each of them. The students form into 4 ASD teams of 5 people each to work on coursework projects spanning over 8 weeks. We have interviewed the course instructors about the students' emotional stability characteristics based on their observations throughout the coursework projects. They provided ratings on each student i 's emotional stability, ES_i , on an 11-point *Likert* scale [26] (0-10 with 0 representing "very poor emotional stability" and 10 representing "very good emotional stability"). The instructor provided ratings are treated as the ground truth in this user study. The distribution of the 20 students' stable state mood values against the number of steps taken to reach equilibrium as calculated by the proposed FCM model is shown in Figure 4(a).

Let i denote a student. Firstly, we study the correlation between the students' stable state mood (M_i) and the number of iteration steps (S_i) taken to reach the equilibrium state according to the proposed FCM model with the instructors' ratings on their emotional stability (ES_i). Figure 4(b) shows the distribution of M_i against ES_i for all i . The correlation coefficient between M_i and ES_i is 0.6996, indicating a strong positive correlation. Figure 4(c) shows the distribution of S_i against ES_i for all i . The correlation coefficient between S_i and ES_i is -0.9383, indicating a very strong negative correlation. Thus, among the two metrics proposed in this

TABLE I
REGRESSION ANALYSIS RESULTS

Constants	Coefficient	Std. Error	p -value
α_1	5.818***	1.543	0.00152
α_2	-0.753***	0.072	8.63×10^{-9}
β	8.781***	1.911	0.00026

Note: ***: $p < 0.01$.

study, S_i shows stronger explanation power for a person's emotional stability than ES_i .

Based on the study results, we investigate the following regression model which can be used to estimate a person's emotional stability based on results produced by the proposed FCM method:

$$ES_i = \alpha_1 M_i + \alpha_2 S_i + \beta. \quad (7)$$

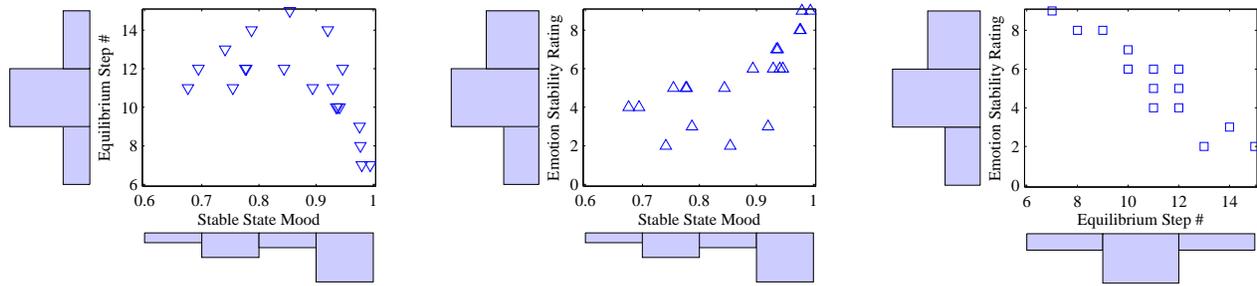
α_1 and α_2 are the linear regression coefficients, whereas β is the y-axis intercept value.

Based on the analysis of the 20 study participants' data, the regression analysis results are shown in Table I. It can be concluded that the stable state mood M_i and the equilibrium steps S_i of the proposed method can be used as explanatory factors for a student i 's emotional stability ES_i in the context of ASD with 99% confidence.

VI. CONCLUSIONS AND FUTURE WORK

In this paper, we proposed a Fuzzy Cognitive Map (FCM) based method for analyzing an important and difficult to measure concept involved in ASD - a developer's emotional stability. We designed an easy-to-use survey questionnaire which can help establish the values of the causal relationship weights to support personalization of the proposed FCM model for each individual user. The FCM affective model is produces two metrics for assessing a given user's emotional stability, which is important to the progress and quality of a software project. They are: 1) the stable state mood value, and 2) the number of iteration steps for the FCM model to reach equilibrium. The two metrics are evaluated through a real-world experiment involving 20 undergraduate students. The results show that the proposed metrics, especially S_i , are strong explanation factors for a user's emotional stability with 99% confidence. A regression model based on the user study has also been presented in the paper to help researchers study the relationships between the two metrics and a person's emotional stability computationally.

From this work, we see a series of interesting future research directions. In this work, we are currently using the basic FCM model. However, there are many sophisticated FCM models, such as [27], which helps reduce human intervention at various stages. In subsequent research, we plan to look into these models to improve the proposed FCM affective model. As the current user study involved only 20 subjects who are all undergraduate students, we refrain from generalizing our findings. In the future, we will conduct larger scale empirical



(a) The distribution of students' stable state mood vs. the number of steps taken to reach values under the proposed FCM model. (b) The distribution of students' stable state mood vs. the number of steps taken to reach values under the proposed FCM model vs. the instructors' ratings on their emotional stability. (c) The number of steps taken to reach equilibrium values vs. the number of steps taken to reach values under the proposed FCM model vs. the instructors' ratings on their emotional stability.

Fig. 4. Empirical Findings.

studies involving subjects with more diverse backgrounds to improve the generalizability of the results obtained.

ACKNOWLEDGEMENTS

This research is supported by the National Research Foundation, Prime Minister's Office, Singapore under its IDM Futures Funding Initiative.

REFERENCES

- [1] A. Cockburn and J. Highsmith, "Agile software development: The people factor," *Computer*, vol. 34, no. 11, pp. 131–133, 2001.
- [2] J. Lin, H. Yu, Z. Shen, and C. Miao, "Studying task allocation decisions of novice agile teams with data from agile project management tools," in *Proceedings of the 29th ACM/IEEE International Conference on Automated Software Engineering (ASE'14)*, 2014, pp. 689–694.
- [3] E. Eldar and Y. Niv, "Interaction between emotional state and learning underlies mood instability," *Nature Communications*, vol. 6, no. 6149, p. doi:10.1038/ncomms7149, 2014.
- [4] Z. Shen, H. Yu, C. Miao, and J. Weng, "Trust-based web-service selection in virtual communities," *Journal for Web Intelligence and Agent Systems (WIAS)*, vol. 9, no. 3, pp. 227–238, 2011.
- [5] Y. Liu, S. Liu, H. Fang, J. Zhang, H. Yu, and C. Miao, "Reprev: Mitigating the negative effects of misreported ratings," in *Proceedings of the 28th AAAI Conference on Artificial Intelligence (AAAI-14)*, 2014, pp. 3124–3125.
- [6] H. Yu, Z. Shen, C. Miao, B. An, and C. Leung, "Filtering trust opinions through reinforcement learning," *Decision Support Systems (DSS)*, vol. 66, pp. 102–113, 2014.
- [7] A. Ortony, G. L. Clore, and A. Collins, *The Cognitive Structure of Emotions*. Cambridge University Press, 1990.
- [8] C. Conati, "Probabilistic assessment of user's emotions in educational games," *International Journal of Applied Artificial Intelligence*, vol. 16, no. 7–8, pp. 555–575, 2002.
- [9] H. Zhang, Z. Shen, X. Tao, C. Miao, B. Li, Ailiya, and Y. Cai, "Emotional agent in serious game (DINO)," in *Proceedings of the 8th International Conference on Autonomous Agents and Multiagent Systems (AAMAS'09)*, 2009, pp. 1385–1386.
- [10] J. Lin, H. Yu, Z. Shen, and C. Miao, "Using goal net to model user stories in agile software development," in *Proceedings of the 15th IEEE/ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing (SNPD'14)*, 2014, pp. 1–6.
- [11] H. Yu, Z. Shen, C. Miao, and A.-H. Tan, "A simple curious agent to help people be curious," in *Proceedings of the 10th International Conference on Autonomous Agents and Multiagent Systems (AAMAS'11)*, 2011, pp. 1159–1160.
- [12] Q. Wu, X. Han, H. Yu, Z. Shen, and C. Miao, "The innovative application of learning companions in virtual singapura," in *Proceedings of the 12th International Conference on Autonomous Agents and Multiagent Systems (AAMAS'13)*, 2013, pp. 1171–1172.
- [13] Y. Cai, Z. Shen, S. Liu, H. Yu, X. Han, J. Ji, M. J. McKeown, C. Leung, and C. Miao, "An agent-based game for the predictive diagnosis of parkinson's disease," in *Proceedings of the 13th International Conference on Autonomous Agents and Multiagent Systems (AAMAS'14)*, 2014, pp. 1663–1664.
- [14] J. Lin, H. Yu, C. Miao, and Z. Shen, "An affective agent for studying composite emotions," in *Proceedings of the 2015 International Conference on Autonomous Agents and Multiagent Systems (AAMAS'15)*, 2015, pp. 1947–1948.
- [15] X. Yu, C. T. Salmon, and C. Leung, "Emotional interactions between artificial companion agents and the elderly," in *Proceedings of the 2015 International Conference on Autonomous Agents and Multiagent Systems (AAMAS'15)*, 2015, pp. 1991–1992.
- [16] X. Yu, C. Miao, C. Leung, and C. T. Salmon, "Modelling composite emotions in affective agents," in *Proceedings of the 2015 IEEE/WIC/ACM International Joint Conference on Web Intelligence and Intelligent Agent Technology (WI-IAT'15)*, 2015.
- [17] S. D. Fraser, F. P. Brooks, Jr., M. Fowler, R. Lopez, A. Namioka, L. Northrop, D. L. Parnas, and D. Thomas, "'No silver bullet' reloaded: Retrospective on 'Essence and accidents of software engineering,'" in *Companion to the 22nd ACM SIGPLAN Conference on Object-oriented Programming Systems and Applications Companion (OOPSLA'07)*, 2007, pp. 1026–1030.
- [18] R. Colomo-Palacios, C. Casado-Lumbreras, P. Soto-Acosta, and Ángel Garca-Crespo, "Using the affect grid to measure emotions in software requirements engineering," *Journal of Universal Computer Science*, vol. 17, no. 9, pp. 1281–1298, 2011.
- [19] B. Kosko, "Fuzzy cognitive maps," *International Journal of Man-Machine Studies*, vol. 24, no. 1, pp. 65–75, 1986.
- [20] A. K. Tsadiras, "Comparing the inference capabilities of binary, trivalent and sigmoid fuzzy cognitive maps," *Information Sciences*, vol. 178, no. 20, pp. 3880–3894, 2008.
- [21] W. Stach and L. Kurgan, "Modeling software development projects using fuzzy cognitive maps," in *Proceedings of the 4th ASERC Workshop on Quantitative and Soft Software Engineering (QSSE'04)*, 2004, pp. 55–60.
- [22] N. Bhatia and N. Kapoor, "Fuzzy cognitive map based approach for software quality risk analysis," *ACM SIGSOFT Software Engineering Notes*, vol. 36, no. 6, pp. 1–9, 2011.
- [23] S. Hossain and L. Brooks, "Fuzzy cognitive map modelling educational software adoption," *Computer and Education*, vol. 51, no. 4, pp. 1569–1588, 2008.
- [24] L. Cao, B. Ramesh, and T. Abdel-Hamid, "Modeling dynamics in agile software development," *ACM Transactions on Management Information Systems (TMIS)*, vol. 1, no. 1, pp. 5:1–5:26, 2010.
- [25] J. Broekens and W.-P. Brinkman, "AffectButton: A method for reliable and valid affective self-report," *International Journal of Human-Computer Studies*, vol. 71, no. 6, pp. 641–667, 2013.
- [26] R. Likert, "A technique for the measurement of attitudes," *Archives of Psychology*, vol. 22, no. 140, pp. 1–55, 1932.
- [27] Y. Miao, "Modelling dynamic causal relationship in fuzzy cognitive maps," in *Proceedings of the 2014 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE'14)*, 2014, pp. 1013–1020.

A Method to Build Bayesian Networks based on Artifacts and Metrics to Assess Agile Projects

Renan Willamy^{*1}, João Nunes^{†1}, Mirko Perkusich^{‡1}, Arthur Freire^{§2}, Renata Saraiva^{¶2}, Hyggo Almeida^{||2}, and Angelo Perkusich^{**2}

¹Paraíba Federal Institute of Education, Science and Technology, Monteiro, Brazil

²Embedded and Pervasive Computing Laboratory, Federal University of Campina Grande, Campina Grande, Brazil

Abstract

Managing software development projects is a complex task because it requires organizing and monitoring several activities. Recently, in order to assist on software projects management, artifact-based models were proposed in the literature. However, the current solutions do not present means to monitor projects health and assist on decisions making. Due to the recent popularization of agile methods, they are the units of study of this research. In this work, we present a method to build artifact and measurement-based models to assess agile projects health. We applied the method to build a generic model based on industries best practice. We defined the models artifacts and metrics based on findings of a literature review and the assistance of an expert. For each models artifact, we applied the Goal-Question-Metric paradigm to define the metrics. Afterwards, from the GQM meta-model, we constructed a Bayesian network. We validated the model with simulated scenarios. Given the successful results, we concluded that the method and model are promising.

Agile Methods; Bayesian Network; Artifact; Metric; Goal-Question-Metric

*renanwillamy2@gmail.com

†lockenunes@gmail.com

‡mirko.perkusich@ifpb.edu.br

§arthur.freire@embedded.ufcg.edu.br

¶renata.saraiva@embedded.ufcg.edu.br

||hyggo@embedded.ufcg.edu.br

**perkusic@embedded.ufcg.edu.br

1 Introduction

Managing software development projects is a complex task, because it is necessary to organize and monitor many activities such as requirements, architecture and project definition, coding, testing and implementation of the product. According to Emam and Koru [7], between 46% and 55% of IT projects fail. According to Boehm et al. [4], the top six reasons for this high failure rate are: incomplete requirements, lack of user involvement, lack of resources, unrealistic expectations, lack of executive support and changing requirements and specifications. Most of these reasons are caused by communication and interaction issues between developers and stakeholders.

Recently, agile methods have become largely adopted by software development industry [27]. Agile methods are focused on the improvement of collaboration between developers and stakeholders in order to improve effectiveness when requirements change.

Managing agile projects is a big challenge, specially in organizations with traditional culture [21]. Besides being flexible to changing requirements and adopting iterative and incremental development, the adoption of agile methods is complex given the context of architecture definition, requirements prioritization, and quality assurance. According to VersionOne [26], 44% of respondents pointed to lack of experience with agile methods as the main cause for agile to fail; 42% to company philosophy or culture at odds with core agile values. As barriers to agile adoption, 44% pointed to the ability to change organizational culture, 35% to not enough personnel with the necessary agile experience, 32% to pre-existing waterfall framework. Therefore, there is a need for resources to assist on the adoption and

continuous improvement of agile methods.

According to Briand et al. [5], it is important to assess a software development process model to define the study objects of a measuring program, and, consequently, the necessary metrics to be collected. In the literature, artifact-based models are proposed to assist on the managing of projects, specially, distributed teams-based projects. Artifacts represent the products related to the software development project such as source code, documentation, specifications e architectural models.

Kuhrmann et al. [16] presented a generic artifact-based model for distributed agile software development teams. Despite the fact that this model is a result of an evolution for centralizing the management of agile distributed teams, some information regarding the project's health, which help on the decision making process, are missing. In our context project health is the current status of the project given its goals (e.g., scope and time restrictions).

These missing information can be collected through software metrics. These metrics allow the measurement, assessment, control and improvement of software artifacts [11]. As a result, it is possible to define a meta-model connecting the artifacts as well as their respective metrics. However, only a meta-model is not enough to assess the artifact according to the collected data.

Recently, Bayesian Networks have become a popular as a mechanism for constructing executable models that are able to deal with uncertainty and assist on the decision making process. It has been applied in several context of software engineering such as risk management [8, 9, 15, 13], quality prediction [1, 14] and process management [19].

In this paper, we aim to complement the work presented by Kuhrmann et al. [16] by presenting a method for constructing artifact-based probabilistic models to assist on the management of agile projects. With the help of a specialist, a generic model (i.e., Bayesian Network) for agile projects was constructed, addressing the main agile projects artifacts and their respective metrics. To define the metrics, we applied the Goal-Question-Metric (GQM) paradigm [2]. Given that the model is a Bayesian Network, its structure facilitates possible modifications to address other artifacts and metrics.

The method and model presented in this paper are limited to the context of projects composed by only one team. To validate them, we used ten simulated scenarios. Based on the results, we concluded that both method and model are promising.

This paper is organized as follows. Section 2 presents an overview of the usage of Bayesian networks in software engineering context. Section 3 presents the method developed to build Bayesian networks based on artifacts and metrics to assess the health of the project. Section 4 presents the generic model build for agile projects. Section 5 the results

of the validation Section 6 presents our conclusions, limitations and future works.

2 Overview of Bayesian Networks applied to Software Engineering

Bayesian networks have been applied to many areas in software engineering such as risk management and product quality prediction. Fan and Yu [8], Fenton et al. [9] and Hu et al. [13] modeled software processes to support risk management. Fan and Yu [8] built a model capable of predicting potential risks, identifying the source of risks, and supporting dynamic resource adjustment. [9] showed how to use a Bayesian network to predict software defects and perform "what if" scenarios. Jeet et al.[15] built a model to estimate the impact of low productivity on the schedule of a software development. They used interviews and historical data to build the model. Hu et al. [13] proposed a risk identification framework for outsourced software projects.

Abouelela and Benedicenti [1] and Jeet et al. [14] modeled software processes to support quality management. Abouelela and Benedicenti [1] built a model to predict a releases rate of defects in a XP project and its duration. Jeet et al. [14] built a model to detect the number of defects in a software development project. The rate of defects and the project manager's judgments are used for predictions and support in managing the number of defects.

Settas et al. [22], Stamelos [23], Stamelos et al. [24] and [19] modeled software processes to support other project management activities. Settas et al. [22] and Stamelos [23] used Bayesian networks to help managerial decision making by modeling software project management anti-patterns. Stamelos et al. [24] modeled the uncertainties of factors to estimate software productivity. Perkusich et al. [19] modeled software processes to support continuous improvement.

Hearty et al. [12] and Perkusich et al. [18] applied Bayesian networks to agile context. Hearty et al. [12] used a Learning Dynamic Bayesian network model to predict project velocity in XP. Perkusich et al. [18] modeled the processes of Scrum projects to detect problems in the team and processes. None of them used artifacts as a base to build the Bayesian networks.

3 The Method

The goal of the method is to assist on the construction of a Bayesian Network-model based on artifacts and metrics to assess the health of agile projects, which consists of their status given their goals. This method consists of five sequential steps: (i) measurement goals definition; (ii) artifacts definition; (iii) metrics definition; (iv) Bayesian Net-

work construction; and (v) validation. Despite being sequential, there are feedback loops between the steps. In step (i), the measuring goals are defined according to the needs. In general, they are related to a project restriction, such as schedule, budget or scope.

In step (ii), development processes artifacts that are related to the measurement goals are defined. An artifact might be associated to more than one measurement goals, and a measurement goal might be associated to more than one artifact, which is a many-to-many relationship. In this step, it is preferable to choose artifacts that are already used in the development process to avoid additional effort on the measurement process.

To execute step (iii), it is necessary to apply the GQM paradigm. Since an artifact may be associated to more than one measurement goal, it is possible to have more than one instance for a given artifact. For each instance, according to its measurement goal, it is necessary to define the goal (i.e., level G), a set of questions (i.e., level Q) for this goal, and metrics (i.e., level M) to be collected in order to measure if the goal associated to the artifact is reached.

In step (iv), the meta-model constructed in step (iii) is transformed in a Bayesian Network. To define the DAG (Directed Acyclic Graph), it is necessary to define a node for each artifact. Afterwards, these nodes need to be decomposed until the metrics associated to them are reached. This can be done by following some steps:

1. For each managing area, define a node;
2. For each goal, define a node and connect it to one or more management areas;
3. For each relationship, an edge is added, in which the edge's endpoint points to the management area;
4. For each artifact, a node is defined and, then, connected to one or more goals;
5. For each relationship, an edge is added, in which the edge's endpoint points to the goal;
6. For each question, define a node and connect it to an artifact. For each relationship, an edge is added, in which the edge's endpoint points to the artifact;
7. For each metric, define a node and connects it to one or more questions;
8. For each metric, an edge is added, in which the edge's endpoint points to the question.

After transforming the GQM meta-model into the DAG, it is necessary to define the probability functions by performing "what if" analysis.

In step (v), it is necessary to validate the Bayesian Network. For this purpose, it is possible to use the Brier score [10] and simulated scenarios [18].

4 Generic Model

To build the generic model, we used the artifacts presented in Kuhrmann [16] and elicited knowledge from an expert. In Section 5, we present the results and the profile of the expert. Moreover, we executed the method presented in Section 3.

The goal of this generic model is to represent an agile project, based on agile best practices, to measure its health. Given that the projects run in different contexts, there might be necessary to modify the model to apply it. Usually, In the context of software process modeling, organizations need to adapt methods and models, so they fit their contexts. However, we believe the generic model presented is robust enough to guarantee minimum modification.

The first step for constructing the model is to define the goals of its measuring. We used the Agile Manifesto [3] to define the measuring goals. In addition to that, we tried to categorized the principles of the Agile Manifest based on projects restrictions: scope, quality, schedule, and cost. We introduce some agile principle as follows:

1. The highest priority is to satisfy the customer through early and continuous delivery of valuable software;
2. Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale;
3. Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale;
4. Continuous attention to technical excellence and good design enhances agility.

By analysing these principles, it is possible to classify 1, 2, and 4 as scope principles; and 3 as a schedule principle. We considered principles related to product quality connected to the scope principles. We did not consider metrics for cost, because they are not directly related to any agile principle.

Afterwards, the artifacts were defined. As scope artifacts, we considered the *Product Backlog*, the *Product Increment*, and the *Tests Report*. Since the model is limited to the project level and a product can be delivered through more than one project (i.e., releases), we also decided to use the *Release Backlog*, which is a subset of the *Product Backlog*. As schedule artifact, we only considered the *Release Burndown*. For each artifact, the GQM template was applied to describe its the measuring goal.

- G1: Analyse the *Release Backlog* for characterizing it regarding the product quality, from the viewpoint of the PO;

- G2: Analyse the *Product Increment* for characterizing it regarding the product satisfaction, from the viewpoint of the PO;
- G3: Analyse the *Tests Report* for characterizing it regarding the tests completeness, from the viewpoint of the PO;
- G4: Analyse the *Release Burndown* for characterizing it regarding the product progress, from the viewpoint of the PO.

Then, for each artifact, we defined a set of questions and metrics. Due to space limitations, we only present the questions for the *Release Backlog*. To define the questions for this artifact, the expert recommended to use the *DEEP* criteria [20]. Given this, we defined four questions: "Is it correctly detailed?", "Is it correctly estimated?", "Does it evolve correctly?", and "Is it correctly ordered?"

For each question, we derived one metric: *item detail level*, *estimation conformity*, and *dynamics of the estimation*. Since these metrics are all subjective, we decided to represent them in 5-point Likert scale. To collect them, it is necessary to answer, respectively, the following questions: "Do you agree that the next iteration's items satisfy the *INVEST* criteria [6]?", "Are the next iteration's items small enough so they be done in one iteration?", and "Does the backlog evolve (i.e., new items added, existing items refined, deleted or reordered) continuously according to the product feedback and changes in the business scenario?"

Since the space is limited, we will only present the assessment method for the metric *item detail level*. Each item allocated for the next iteration needs to be assessed according to the *INVEST* criteria. This criteria assess six factors: (i) independence, (ii) negotiability, (iii) value, (iv) if it is estimable (v) size, and (iv) testability. It's ideal that all items are independent; their scope had been already discussed between the development team and the client; they have business value; the developers have enough knowledge to estimate them; and they're small enough so they can be done in one iteration, as well as tested considering non-functional requirements (e.g., performance, usability, and security). For this purpose, it is necessary to assess each backlog item according to these criteria in a 5-point scale (i.e., "1" for Very Low, and "5" for Very High). Afterwards, the results need to be aggregated and mapped into the *item detail level* metric. For doing so, we used the *WMIN* function.

Regarding the question "Is it correctly ordered?", the metrics were defined according to the criteria *DIVE* [20]. These metrics are: *backlog items dependency*, *risks*, *estimation conformity*, and *business value*. Since these metrics are all subjective, we decided to represent them in 5-point Likert scale. To collect these metrics, it is necessary to an-

swer, respectively, the following questions: "Do you consider the backlogs' items technical dependency?", "Do you consider the backlogs' items technical and business risks?", "Are next iteration's items estimated with points or ideal days?", and "Do you consider the business value, in the contexts of the user, schedule, and organization?" For the *business value* metric, it is possible to use the numeric methods presented by Thatte [25]. However, the assessment of their usage wasn't addressed in this study. After defining the set of questions and metrics for each artifact identified, we built a GQM meta-model, which was later transformed into the DAG. We present the DAG in Figure 1.

To calibrate the probability functions, we used the knowledge of one expert, which has over five years of experience as an agile project manager. For each child-node, we defined a questionnaire composed by a set of combinations of its parent nodes. Due to space limitations, we limit ourselves to state that, to generate the probability functions, we applied the approach proposed by Laitila [17].

5 Validation

We validated the model and the procedure in ten simulated scenarios, defined according to the expert availability. To due space limitations, we only present the results of one scenario. This scenario (i) describes an experienced and efficient development team, in which the manager is ineffective and absent.

For each scenario, according to their contexts, the values for the input nodes (i.e., nodes with no parents) were defined by the specialist. Furthermore, for the nodes related to the goals and artifacts, the specialist defined the expected values. For this scenario, the expected results were:

- Scope: central tendency between *Very Low* and *Low*.
- Time: central tendency between *Very Low* and *Low*.
- G1: central tendency between *Very Low* and *Low*.
- G2: central tendency between *Medium* and *Low*.
- G3: central tendency between *Medium* and *Low*.

We present the results for this scenario in Table 1. These results match the expected results we defined before starting the validation process. Given this, we concluded both method and generic model are promising. However, the number of scenarios we defined for validating the method and model are not enough to conclude they would be useful in the context of real projects.

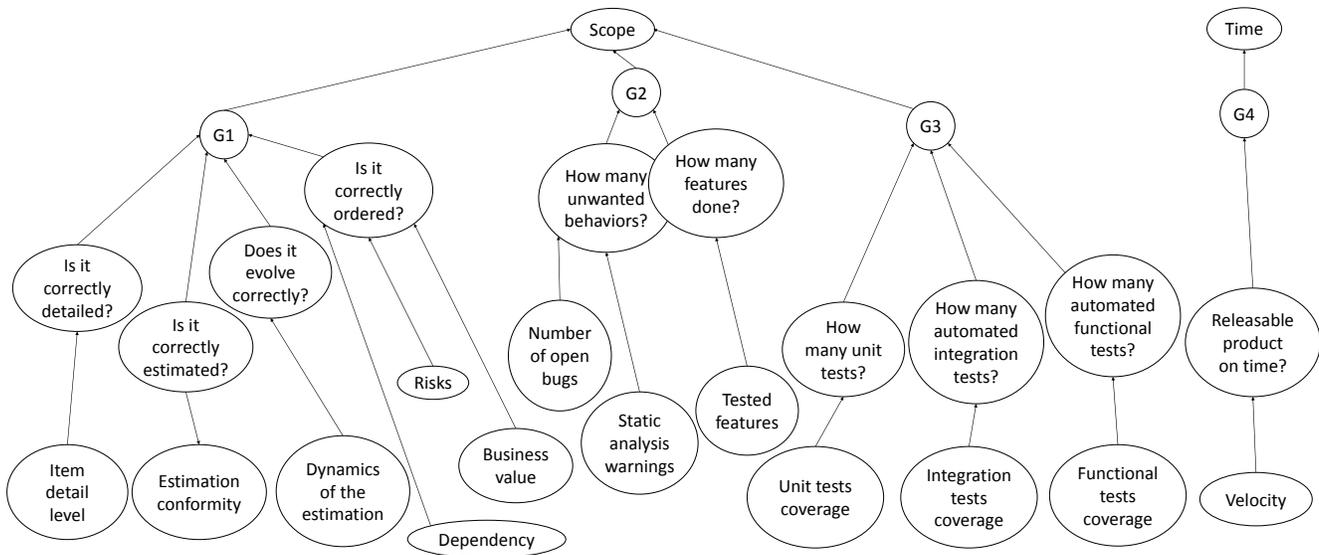


Figure 1. Complete directed acyclic graph for the Bayesian network.

Node	States				
	VL	L	M	H	VH
Scope	56%	44%	0	0	0
Time	0	100%	0	0	0
G1	100%	0	0	0	0
G2	0	61%	39%	0	0
G3	0	70%	30%	0	0

Table 1. Scenario results for highest level nodes in the model

6 Conclusions

In this paper, we presented a method to construct Bayesian networks based in artifacts and metrics to assist in agile project management. The method consists of five steps: (i) measurement goals definition; (ii) artifacts definition; (iii) metrics definition; (iv) Bayesian Network construction; and (v) validation. The purpose of the method is to measure agile projects' health.

The method was applied to construct a generic model considering the most popular artifacts and metrics in the industry. The model's measurement goals were defined according to agile principles, and it was constructed with the help of an expert. We validated the model through two simulated scenarios with positive results. Thus, we concluded that the results are promising.

This study limitations are related to the quantity of simulated scenarios in which we performed the validation. In addition to that, we only collected knowledge from one expert. However, both limitations are due to the unavailability

of industry experts to collaborate in the study during the required period.

For future works, we pretend to perform a systematic review to identify metrics, consult other specialists for constructing the model, perform empirical case studies to evaluate the method and the models constructed in industry context, and extend the method to consider projects with multiple distributed teams.

Acknowledgments

This work was financially supported by Paraba Federal Institute of Education, Science and Technology, Campus Monteiro.

References

- [1] M. Abouelela and L. Benedicenti. Bayesian network based xp process modelling. *International Journal of Software Engineering and Applications*, 1(3):1–15, 2010.
- [2] V. R. Basili and D. M. Weiss. A methodology for collecting valid software engineering data. *IEEE Transactions on Software Engineering*, SE-10(6):728–738, Nov 1984.
- [3] K. Beck, M. Beedle, A. van Bennekum, A. Cockburn, W. Cunningham, M. Fowler, J. Grenning, J. Highsmith, A. Hunt, R. Jeffries, J. Kern, B. Marick, R. C. Martin, S. Mellor, K. Schwaber, J. Sutherland, and D. Thomas. Manifesto for agile software development, 2001. Accessed: 29th February 2016.

- [4] B. Boehm. Software engineering is a value-based contact sport. *IEEE Softw.*, 19(5):95–96, Sept. 2002.
- [5] L. C. Briand, C. M. Differding, and H. D. Rombach. Practical guidelines for measurement-based process improvement. *Software Process: Improvement and Practice*, 2(4):253–280, 1996.
- [6] M. Cohn. *Succeeding with Agile: Software Development Using Scrum*. Addison-Wesley Professional, 1st edition, 2009.
- [7] K. E. Emam and A. G. Koru. A replicated survey of it software project failures. *IEEE Software*, 25(5):84–90, Sept 2008.
- [8] C.-F. Fan and Y.-C. Yu. Bbn-based software project risk management. *Journal of Systems and Software*, 73(2):193–203, Oct. 2004.
- [9] N. Fenton, P. Krause, and M. Neil. Software measurement: uncertainty and causal modeling. *Software, IEEE*, 19(4):116–122, 2002.
- [10] N. Fenton and M. Neil. *Risk assessment and decision analysis with Bayesian networks*. CRC Press, 2012.
- [11] K. A. M. Ferreira, M. A. S. Bigonha, R. S. Bigonha, L. F. O. Mendes, and H. C. Almeida. Identifying thresholds for object-oriented software metrics. *J. Syst. Softw.*, 85(2):244–257, Feb. 2012.
- [12] P. Hearty, N. Fenton, D. Marquez, and M. Neil. Predicting project velocity in xp using a learning dynamic bayesian network model. *IEEE Transactions on Software Engineering*, 35(1):124–137, Jan 2009.
- [13] Y. Hu, X. Mo, X. Zhang, Y. Zeng, J. Du, and K. Xie. Intelligent analysis model for outsourced software project risk using constraint-based bayesian network. *Journal of Systems and Software*, 7(2):440 – 449, 2012.
- [14] K. Jeet, N. Bhatia, and R. S. Minhas. A bayesian network based approach for software defects prediction. *SIGSOFT Softw. Eng. Notes*, 36(4):1–5, Aug. 2011.
- [15] K. Jeet, N. Bhatia, and R. S. Minhas. A model for estimating the impact of low productivity on the schedule of a software development project. *SIGSOFT Softw. Eng. Notes*, 36(4):1–6, Aug. 2011.
- [16] M. Kuhrmann, D. M. Fernández, and M. Gröber. Towards artifact models as process interfaces in distributed software projects. In *Proceedings of the 2013 IEEE 8th International Conference on Global Software Engineering, ICGSE '13*, pages 11–20, Washington, DC, USA, 2013. IEEE Computer Society.
- [17] P. Laitila. Improving the use of ranked nodes in the elicitation of conditional probabilities for bayesian networks. Master’s thesis, Aalto University, Finland, 2013.
- [18] M. Perkusich, H. O. de Almeida, and A. Perkusich. A model to detect problems on scrum-based software development projects. In *Proceedings of the 28th Annual ACM Symposium on Applied Computing*, pages 1037–1042. ACM, 2013.
- [19] M. Perkusich, G. Soares, H. Almeida, and A. Perkusich. A procedure to detect problems of processes in software development projects using bayesian networks. *Expert Systems with Applications*, 42(1):437 – 450, 2015.
- [20] R. Pichler. *Agile Product Management with Scrum: Creating Products that Customers Love (Adobe Reader)*. Addison-Wesley Professional, 2010.
- [21] K. Schwaber and J. Sutherland. Guia do scrum. <http://www.scrumguides.org/docs/scrumguide/v1/Scrum-Guide-Portuguese-BR.pdf>, 2015. Accessed: 17th March 2016.
- [22] D. Settas, S. Bibi, P. Sfetsos, I. Stamelos, and V. Geroiannis. Using bayesian belief networks to model software project management antipatterns. In *Software Engineering Research, Management and Applications, 2006. Fourth International Conference on*, pages 117–124, 2006.
- [23] I. Stamelos. Software project management anti-patterns. *Journal of Systems and Software*, 83(1):52 – 59, 2010.
- [24] I. Stamelos, L. Angelis, P. Dimou, and E. Sakellaris. On the use of bayesian belief networks for the prediction of software productivity. *Information and Software Technology*, 45(1):51 – 60, 2003.
- [25] S. Thatte. Agile prioritization: a comprehensive and customizable et simple and practical method. <https://goo.gl/97POHR/>, 2014. Accessed: 17th March 2016.
- [26] VersionOne. 9th annual state of agile development survey results. <http://www.versionone.com/pdf/state-of-agile-development-survey-ninth.pdf>, 2015. Accessed: 2015-05-05.
- [27] Y. Zhang and S. Patel. Agile model-driven development in practice. *IEEE Software*, 28(2):84–91, March 2011.

CPDScorer: Modeling and Evaluating Developer Programming Ability across Software Communities

Weizhi Huang, Wenkai Mo, Beijun Shen[‡], Yu Yang, Ning Li

School of Electronic Information and Electrical Engineering

Shanghai Jiao Tong University, Shanghai, China

Email: zhizi1993321@163.com, jirachikai@126.com, [‡]bjshen@sjtu.edu.cn, 13272436992@163.com, lnkkk1994@163.com

Abstract—Since developer ability is recognized as a determinant of better software project performance, it is a critical step to model and evaluate the programming ability of developers. However, most existing approaches require manual assessment, like 360 degree performance evaluation. With the emergence of social networking sites such as StackOverflow and Github, a vast amount of developer information is created on a daily basis. Such personal and social context data has huge potential to support automatic and effective developer ability evaluation. In this paper, we propose CPDScorer, a novel approach to modeling and scoring the programming ability of developer through mining heterogeneous information from both Community Question Answering (CQA) sites and Open-Source Software (OSS) communities. CPDScorer analyzes the answers posted in CQA sites and evaluates the projects submitted in OSS communities to assign expertise scores to developers, considering both the quantitative and qualitative factors. When modeling the programming ability of developer, a programming ability term extraction algorithm is also designed based on set covering. We have conducted experiments on StackOverflow and Github to measure the effectiveness of CPDScorer. The results show that our approach is feasible and practical in user programming ability modeling. In particular, the precision of our approach reaches 80%.

Index Terms—Github; StackOverflow; Programming Ability Modeling; Developer Ability Evaluation.

I. INTRODUCTION

Software developers participate in a diversity of software communities simultaneously, such as Community Question Answering (CQA) sites and Open-Source Software (OSS) communities. For example, in StackOverflow¹, developers post questions to seek for help from other peers, and they are also able to provide valuable answers for others; in Github², they collaboratively develop open-source softwares by committing code to software repositories. These software communities create and store a vast amount of developer information on a daily basis. Thus they provide huge potential to model and evaluate the ability of developer effectively, which plays an important role in developer recommendation, staff training as well as software project planning.

Some approaches have been proposed to target the problem of developer ability modeling in software communities [1] [2]. However, these approaches are restrict to a single community.

[‡] Corresponding Author

¹<http://stackoverflow.com/>

²<https://github.com/>

DOI reference number: 10.18293/SEKE2016-012

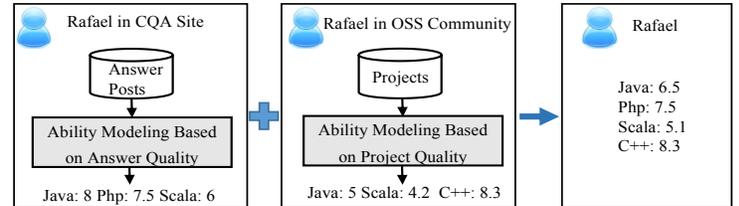


Figure 1: An Example of Developer Ability Modeling across Two Software Communities

For instance, in CQA sites, user ability can be reflected by his or her answer quality. Generally, users who provide high quality answers are more likely to own high expertise in specified subjects. The voting score of each answer is a kind of measure of answer quality as it is given by users according to their satisfaction towards the answer. Similarly, in OSS communities, developers who provide projects with high quality tend to be with a high ability in programming. In this paper, we consider developer ability modeling across software communities for more comprehensive and precise evaluation results. Figure 1 shows a simple example. Developer Rafael participates in both a CQA site and an OSS community. He is assigned ability scores under different programming skills in both two communities, and the ability scores from these two communities produce his final ability scores.

However, it is difficult to model and evaluate the ability of developer crossing communities as we need to link users between different communities first. Besides, integrating heterogeneous data of a developer from different communities is challenging. Moreover, how to derive the concrete programming ability terms to describe the programming ability of developer is also a problem. In this paper, we propose a novel approach CPDScorer to address these challenges. CPDScorer integrates one CQA site with one OSS community to corporately assess the programmer’s ability from both answer quality analysis and project code evaluation. Specifically, CPDScorer models the programming ability scores of developer under various programming skills. To model the programming ability, an algorithm based on set covering is proposed to extract a set of programming ability terms. Then every answer and project are labeled with an ability term and scored based on their quality. Given answer-based ability scores in CQA site and code-based ability scores in OSS

community under different ability terms, we combine these scores by taking a weighted sum of them as the final ability scores of developers.

This paper makes the following contributions: (1) We propose an approach to modeling and evaluating the programming ability of developer crossing two software communities. Specifically, we combine answer quality evaluation in a CQA site and code quality analysis in an OSS community to model the ability of developer more comprehensively. (2) We design a programming ability term extraction algorithm based on set covering to model ability scores and ability terms jointly.

II. RELATED WORK

A. Developer Ability Modeling in Communities

To model and evaluate the ability of developer in communities, several methods have been proposed. Zhang et al. proposed a measure called Z-score that combined user's asking and replying patterns to rate the expertise of user [3]. Liu et al. proposed CQARank to model each user's ability as expertise scores under various topics by combining textual content learning with link analysis in StackOverflow [1]. Venkataramani and Asadullah modeled the expertise of developers in a target domain by mining their activities in different open-source projects [2]. John and Gail presented an empirical evaluation of two approaches for determining the implementation expertise of developer from the data in source and bug projects [4]. All these methods mentioned above only analyze information from a single software community, and can not utilize data across different communities.

B. Answer Quality Evaluation in CQA Sites

There are some researches to evaluate the answer quality in CQA sites. Jeon et al. presented a model to predict the quality of answers based on a set of non-textual features extracted from answers, such as click counts, answerers acceptance ratio and answer length [5]. Agichtein et al. casted the problem of answer quality ranking as a binary classification problem and proposed a classification framework of estimating answer quality based on content-based features and usage-based features [6]. Shah et al. used 13 different criteria to assess the overall quality of answer and expanded the prediction model by extracting several features of the questions, the answers, and the users who provided them [7]. When modeling the developer's ability by analyzing the answer quality in CQA sites, our work borrows the idea in [5].

C. Open-Source Project Quality Analysis

With the emergence of open-source projects, measuring and evaluating their quality has attracted a lot of attention. Jarczyk et al. developed two metrics of quality for the open-source projects in Github based on both the project popularity and how fast the reported issues were solved, and analyzed the influence of collected attributes describing project and developer on quality [8]. Different from the above approach, we focus on source code evaluation for project quality. Stamelos et al. measured quality characteristics of Linux applications

by a kind of software measurement tool and compared the results with the industrial standard proposed by the tool [9]. Barkmann et al. developed tools to collect code metric data from projects and described the statistical significance of individual metrics [10]. Chawla et al. implemented five metrics such as lines of code and cyclomatic complexity to analyze a set of java programs as to judge their performance with respect to the metrics [11]. We borrow the idea from [11] to analyze the quality of project source code.

III. APPROACH

Our proposed approach CPDScorer can model and evaluate the programming ability of developer automatically, by analyzing data from one CQA site and one OSS community.

A. Approach Overview

The overview of CPDScorer approach is illustrated in Figure 2. It is composed of five steps: (1) *Identity Linkage* links users between a CQA site and an OSS community. (2) *Programming Ability Term Extraction* extracts a set of programming ability terms based on set covering. Meanwhile, each answer or project is assigned an ability term by its topic. (3) *Answer-Based Ability Scoring* analyzes the quality of each answer in the CQA site, and assigns ability scores to the question repliers by ability terms of answers. (4) *Code-Based Ability Scoring* evaluates the quality of each software project in OSS community using code analysis technology, and scores the developers by specific project ability terms. (5) *Ability Score Composing* takes a weighted sum of answer-based ability scores and code-based ability scores as the final programming ability scores of developers.

B. Identity Linkage across Software Communities

There are already several methods proposed to link users in different software communities [12] [13]. However, these methods only considered username and email of a user while ignoring other important information like programming skills. As we need to link users between a CQA site and an OSS community, we adopt our novel tagging-based approach TBIL [14]. TBIL first extracts three kinds of features from usernames, user skills and user concerned topics from the text information of users. Based on these features, it then applies a machine learning method, Decision Tree, to obtain probabilities for every two users in different communities, which represents how likely they refer to a same person. Finally, based on these probabilities, we use a heuristic greedy matching algorithm to link these users with one-to-one constraints.

C. Programming Ability Term Extraction

After linking developers between a CQA site and an OSS community, we then need to define a set of programming ability terms to specifically describe the programming ability of developers. In some software communities, software documents are labeled with tags. These tags can be treated as ability terms because they all refer to terms about software engineering. We propose a programming ability term extraction algorithm

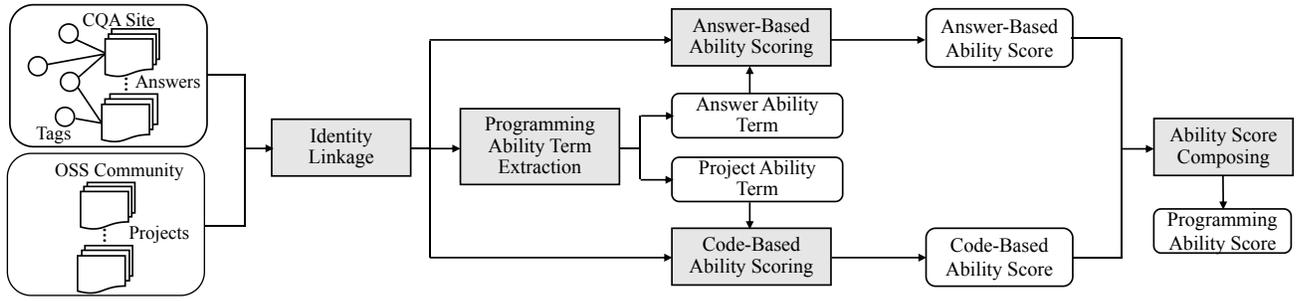


Figure 2: Overview of CPDScore Approach

based on set covering to derive ability terms from tags. The algorithm takes a candidate programming ability term set, and a set of labeled documents (e.g., labeled answers, labeled projects, etc.) as input. It first filters out tags with frequencies less than a threshold and sorts the rest tags by their frequencies in descending order. Next, for each labeled document in the document set, if any tag of the document equals a certain candidate ability term, the ability term will be contained in the final ability term set and this document is set as covered. Finally, if the percentage of covered documents among all the documents (covering rate) reaches 90% or higher, we output the final ability term set. Otherwise, the threshold will be automatically adjusted until the covering rate is higher than 90%.

To perform the algorithm, we first need to construct a candidate ability term set. We take the set of tags in a community as the initial ability term set. Then we remove tags which are unrelated with programming ability and form a new set of tags as the candidate ability term set. For example, the tags ‘excel’ and ‘firefox’ can not reflect the programming ability of developer. As the removing process is done manually, we will clean some low frequency tags before the process to improve performance. When constructing the candidate ability term set in two different communities, there are three cases: (1) If both two communities do not have the tagging system, then we can choose the tags in StackOverflow because StackOverflow has a relatively mature tagging system with more than 38,000 diverse tags. (2) If only one community has the tagging system, tags in this community are used. (3) If both two communities have the tagging system, we select the intersection of tags in these two communities.

The detailed description of the ability term extraction procedure is shown in Algorithm 1. It takes a candidate ability term set S , and a labeled document set D as input. The algorithm first initializes the covering rate μ to 0 and the threshold θ to 5000. In each repeat, θ will be gradually reduced by 300 until μ is higher than 90%. Then our algorithm filters out some tags of which the frequencies are less than the threshold and sorts tags by their frequencies in descending order (Lines 1-6). For each labeled document, if any of its tags equals a certain candidate ability term, the status of this document is set as covered. After traversing every document, we calculate and update the covering rate (Lines 7-16). Finally, the algorithm

Algorithm 1 Programming Ability Term Extraction

Input:

candidate programming ability term set S ;
 labeled document set $D = \{d_1, d_2, \dots, d_N\}$;

Output:

programming ability term set H ;

- 1: initialize covering rate μ to 0
 - 2: initialize threshold θ to 5000
 - 3: **repeat**
 - 4: automatically adjust θ
 - 5: select the tags in S whose frequencies are larger than θ to form a new candidate set $T = \{t_1, t_2, \dots, t_M\}$
 - 6: sort T by the frequencies of tags in descending order
 - 7: **for** each d_n in D **do**
 - 8: **for** each t_i in T **do**
 - 9: **if** one of the tags in d_n equals t_i **then**
 - 10: set the status of d_n as covered
 - 11: put t_i in H
 - 12: **end if**
 - 13: **end for**
 - 14: **end for**
 - 15: calculate and update μ
 - 16: **until** μ is larger than 90%
 - 17: **return** H ;
-

outputs the final ability term set (Line 17).

Furthermore, we manually divide the programming ability terms into five categories, including ability terms about programming language (such as java, php, etc.), database (such as mysql, mongodb, etc.), framework (such as asp.net, spring, etc.), library (such as jquery, backbone.js, etc.) and others (such as Android, etc.).

D. Answer-based Ability Scoring

High quality answers are expected to reflect that the replier has a good knowledge of the specific programming domain (described by the programming ability term) the answers belong to. So, the first sub-step is to assign the appropriate ability term to every answer by matching its tags with terms in the programming ability term set. However, answers in some CQA sites are not labeled with any tags. For such answers, we extract keywords from the content of each answer using

TextRank algorithm [15]. And we match these keywords with programming ability terms to select a most representative term for the answer.

Then the second sub-step is to analyze the quality of each answer in CQA site, and work out the ability scores of the replier. In most CQA sites, answers can be voted up or down considering their quality. Users can also comment on them. We extract features determining answer quality from each answer post, such as answer length, the number of upvotes, and the number of downvotes. After scoring each answer based on its features, the ability score for developer u under a specific programming ability term t is the average score of answers posted by the developer of which the ability term is just t , which is defined as:

$$SEScore(u, t) = \frac{\sum_{ans \in Answer(u)} H(ans, t) AScore(ans)}{\sum_{ans \in Answer(u)} H(ans, t)} \quad (1)$$

where $AScore(ans)$ denotes the evaluation score of answer, $Answer(u)$ represents all the answers posted by the developer u and $H(ans, t)$ is an indicator function, returning 1 if the ability term of answer is t .

E. Code-based Ability Scoring

To model the ability scores of developer based on project quality, we need to obtain the ability term for every project first. As some OSS communities do not have a tagging system, we extract keywords from the description files of projects like README files in Github using TextRank algorithm [15]. The extracted keywords will later be used to match the ability terms for the project ability term. Then we use static code analysis tool Understand to extract code metrics (e.g., CountLineCode, AvgLine, CountPath, etc.) and score projects with code metrics.

Given the score of one project p a developer u involved in, we define Equation (2) to calculate the ability score $GEScore(u, t)$ of the developer under a specific project term t , where $I(p, t)$ is an indicator function, returning 1 if the ability term of project is t . We use $P(u)$ to denote all the projects that developer u are involved in and we define all the developers in the project as $U(p)$. $PScore(p)$ denotes the evaluation score of project p . The contribution that developer u makes to project p is defined as $Cont(u, p)$, which is provided in our dataset. If the contribution of developers is not offered, we can measure it by their commit frequency in the project.

$$GEScore(u, t) = \sum_{p \in P(u)} I(p, t) PScore(p) \frac{Cont(u, p)}{\sum_{u \in U(p)} Cont(u, p)} \quad (2)$$

The project score is allocated to developers according to their contribution to the project. The more contribution they make to the project, the higher score they are allocated. For the developer's ability score under a certain skill t , we sum up all the scores of projects with the target ability term t that the developer is involved in.

<https://scitools.com/>

F. Ability Score Composing

Given a target developer u , *Ability Score Composing* combines the two ability scores along with their programming ability term t produced by *Answer-Based Ability Scoring* and *Code-Based Ability Scoring* into a unified expertise score. We define the final expertise score of developer u in programming skill t denoted by $EScore(u, t)$ as follows:

$$EScore(u, t) = \alpha \times SEScore(u, t) + \beta \times GEScore(u, t) \quad (3)$$

where $SEScore(u, t)$ and $GEScore(u, t)$ are the ability scores of programming skill t evaluated by the two scoring components respectively, and $\alpha, \beta \in [0, 1]$ are weights assigned to the two scores.

IV. EXPERIMENTS AND RESULTS

In this section, we conduct experiments to validate the performance of CPDScorer. We first present our experimental settings and then analyze the results of experiments.

A. Experimental Settings

As StackOverflow is one of the most famous CQA sites and Github is a representative OSS community, we select them to conduct the experiments. The data of StackOverflow and Github is before January, 2015. There are totally 3,106,381 posts including 255,401 answer posts, and 32,207 tags. Besides, there are totally 5791 linked developers involved in 133,895 projects. We extract 305 programming ability terms from the tags in StackOverflow. And the extracted answer features and code metrics are shown in Table I and Table II. There are 5,230 answers and 2,500 projects in training data rated by five master students whose major is computer science on scale 1 to 10. As CPDScorer can model the developer's ability scores under each ability term, we select the top-10 developers in each domain by their ability scores. Then, we ask another five students to evaluate and rate our results by giving 'yes' or 'no' to show whether they are satisfied with the results. Specifically, they will investigate the detailed profiles of the corresponding developers in both StackOverflow and Github, including the quality of their answers, the quality of their projects, their activity of contribution and other related information, to determine whether the results are reasonable. The precision of the CPDScorer approach is defined as the following equation, where M represents the number of 'yes' and N denotes the total number of results (the number is 10 in our experiments).

$$Precision = \frac{M}{N} \quad (4)$$

B. Developer Ability Modeling Using Different Regression Methods

We adopt two widely used machine learning methods: Linear Regression and Regression Tree M5P [16], and separately apply these two methods to both answer features and code metrics. Thus there are four combinations of methods, as shown in Table III. To compare the final results of ability scores using different combinations of methods for their effectiveness, we

TABLE I: LIST OF FEATURES

Features	Description	Coefficients
Answer Length	The length of the answer	0.0023
Answer Vote Score	The score of the answer is calculated for based on the number of upvotes and downvotes of the answer	0.0781
Comment Count	The number of comments to the answer	0.0113
Answer Acceptance	Whether the answer is accepted by the question owner is a direct feedback on the quality of the answer	0.2388
Number of DownVote	The number of the DownVote the answer receives, which shows the answer is useless	-0.019
Number of UpVote	The number of the UpVote the answer receives, which shows the answer is useful	0.0041
Number of Answers	The number of answers for the given programming topic	0.0038

TABLE II: LIST OF CODE METRICS

Code Metrics	Description	Coefficients
AvgCyclomatic	Average cyclomatic complexity for all nested functions or methods	-3.9903
AvgCyclomaticModified	Average modified cyclomatic complexity for all nested functions or methods	4.8879
AvgCyclomaticStrict	Average strict cyclomatic complexity for all nested functions or methods	-0.6188
AvgLineBlank	Average number of blank for all nested functions or methods	-0.4428
AvgLineCode	Average number of lines containing source code for all nested functions or methods	0.8015
CountDeclFunction	Number of functions	0.7291
CountLineBlank	Number of blank lines	-0.845
CountLineCode	Number of lines containing source code	-1.6516
CountLineCodeExe	Number of lines containing executable source code	-0.4279
CountStmtExe	Number of executable statements	9.4282
Cyclomatic	Cyclomatic complexity	0.4577
SumCyclomaticStrictv	Sum of strict cyclomatic complexity of all nested functions or methods	-7.4112

randomly select three ability terms from each ability category and pick the top-10 developers with each ability term to be rated by those five students every time. And the process will be repeated five times. Furthermore, we take the average precision of score results under all ability terms from a specific category as precision of the category. However, we exclude the ‘other’ category on account that the number of developers ranking top can’t reach 10 for most ability terms from that category.

Figure 3 depicts the results. In Figure 3, the horizontal axis represents the combinations of different methods and the vertical axis denotes their respective precisions in the four categories. From the results, we can see the combination LR×M5P achieves the best performance with the precision of 80% among the four combinations in all categories followed by M5P×M5P while LR×M5P has the worst precision. Thus in our experiment, we apply the M5P to answer features and Linear Regression to code metrics.

TABLE III: COMBINATION OF DIFFERENT METHODS

	Linear Regression for AF	M5P for AF
M5P for CM	M5P×LR	M5P×M5P
Linear Regression for CM	LR×LR	LR×M5P

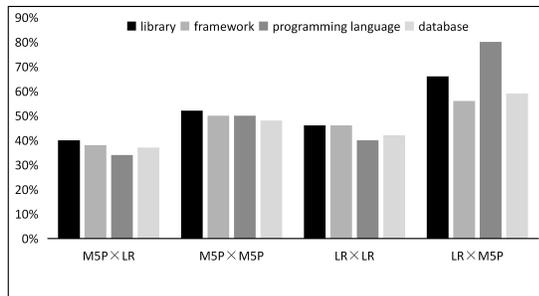


Figure 3: Results of Different Combinations of Methods

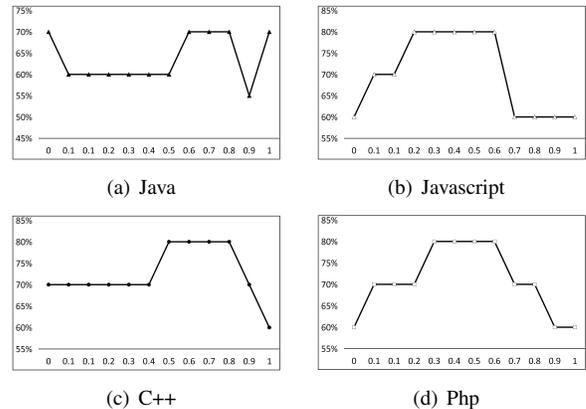


Figure 4: Precisions at Different Values of α

C. Features Contributions

When applying M5P to answer features and Linear Regression to code metrics, it has the best performance. Thus only the coefficients of answer features for M5P and coefficients of code metrics for Linear Regression are presented and discussed.

From Table I, it is not surprising the features ‘Answer Vote Score’ and ‘Answer Acceptance’ have a significant positive effect on the outcome quality scores, while ‘Number of Downvote’ has a negative effect. When the questioner accepts one answer, it means the answer works for him or her well. The larger the ‘Answer Vote Score’ is, the higher value the answer will be thought of. Conversely, ‘Number of Downvote’ suggests that users consider the answer is useless.

As shown in Table II, more than half features have negative impacts on the quality of project source code. Most of them do not affect the performance by much. However, the features ‘AvgCyclomatic’, ‘CountStmtExe’, and ‘SumCyclomaticStrict’ have significant contribution to the code quality.

D. Parameter Sensitivity Analysis

In this experiment, we vary the weights α and β assigned to ability scores from the two communities to understand the impact of varying their values on the precision. The sum of α and β is always 1.0, so we just adjust the value of α . We start by initializing α to 0. Then, we incrementally increase the value of α by 0.1 until it reaches 1. For each combination of α and β , we evaluate the top-10 developers. As the number of ability terms about programming language accounts for nearly half of all ability terms, we focus on analysing scores under programming languages. We randomly select four programming language, such as java, javascript, c++ and php.

Figure 4 illustrates the precisions at different values of α . In Figure 4, horizontal axis denotes different values of α , ranging from 0 to 1. The values in vertical axis reflect the precision of corresponding α . The precision of CPDScorer remains quite stable across a wide range of parameters α . The best precision is achieved among the four ability terms when the value of α becomes 0.6. As a result, we choose the value 0.6 for α and the value 0.4 for β in our work.

V. CONCLUSION AND FUTURE WORK

In this paper, we propose a novel approach called CPDScorer for modeling and evaluating the programming ability of developers crossing two communities. CPDScorer considers both answer quality about programming topics in a CQA site and project source code quality in an OSS community. A programming ability term extraction algorithm is also designed to label every answer and project with an ability term. Two famous communities, StackOverflow and Github, are selected to validate the feasibility of our approach.

To evaluate the programming ability of developers more accurately, there are still some aspects for improvement in the future: (1) When modeling the developer’s ability by estimating their answer posts, we will extract more features

such as the answer comment content. In particular, the profile of the replier may play an important role in answer quality. (2) When modeling the programming ability in OSS communities, we can explore other factors such as the commit messages of developer for project quality evaluation.

ACKNOWLEDGEMENT

This research is supported by 973 Program in China (Grant No. 2015CB352203) and National Natural Science Foundation of China (Grant No. 61472242).

REFERENCES

- [1] L. Yang, M. Qiu, S. Gottipati, F. Zhu, J. Jiang, H. Sun, and Z. Chen, “Cqrank: jointly model topics and expertise in community question answering,” in *Proceedings of the 22nd ACM international conference on Conference on information & knowledge management*, pp. 99–108, ACM, 2013.
- [2] R. Venkataramani, A. Gupta, A. Asadullah, B. Muddu, and V. Bhat, “Discovery of technical expertise from open source code repositories,” in *Proceedings of the 22nd international conference on World Wide Web companion*, pp. 97–98, International World Wide Web Conferences Steering Committee, 2013.
- [3] J. Zhang, M. S. Ackerman, and L. Adamic, “Expertise networks in online communities: structure and algorithms,” in *Proceedings of the 16th international conference on World Wide Web*, pp. 221–230, ACM, 2007.
- [4] J. Anvik and G. C. Murphy, “Determining implementation expertise from bug reports,” in *Mining Software Repositories, 2007. ICSE Workshops MSR’07. Fourth International Workshop on*, pp. 2–2, IEEE, 2007.
- [5] J. Jeon, W. B. Croft, J. H. Lee, and S. Park, “A framework to predict the quality of answers with non-textual features,” in *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, pp. 228–235, ACM, 2006.
- [6] E. Agichtein, C. Castillo, D. Donato, A. Gionis, and G. Mishne, “Finding high-quality content in social media,” in *Proceedings of the 2008 International Conference on Web Search and Data Mining*, pp. 183–194, ACM, 2008.
- [7] C. Shah and J. Pomerantz, “Evaluating and predicting answer quality in community qa,” in *Proceedings of the 33rd international ACM SIGIR conference on Research and development in information retrieval*, pp. 411–418, ACM, 2010.
- [8] O. Jarczyk, B. Gruszka, S. Jaroszewicz, L. Bukowski, and A. Wierzbicki, “Github projects. quality analysis of open-source software,” in *Social Informatics*, pp. 80–94, Springer, 2014.
- [9] I. Stamelos, L. Angelis, A. Oikonomou, and G. L. Bleris, “Code quality analysis in open source software development,” *Information Systems Journal*, vol. 12, no. 1, pp. 43–60, 2002.
- [10] H. Barkmann, R. Lincke, and W. Lowe, “Quantitative evaluation of software quality metrics in open-source projects,” in *Advanced Information Networking and Applications Workshops, 2009. WAINA’09. International Conference on*, pp. 1067–1072, IEEE, 2009.
- [11] M. K. Chawla and I. Chhabra, “Implementing source code metrics for software quality analysis,” in *International Journal of Engineering Research and Technology*, vol. 1, ESRSA Publications, 2012.
- [12] S. Liu, S. Wang, F. Zhu, J. Zhang, and R. Krishnan, “Hydra: Large-scale social identity linkage via heterogeneous behavior modeling,” in *Proceedings of the 2014 ACM SIGMOD international conference on Management of data*, pp. 51–62, ACM, 2014.
- [13] E. Kouters, B. Vasilescu, A. Serebrenik, and M. G. van den Brand, “Who’s who in gnome: Using lsa to merge software repository identities,” in *Software Maintenance (ICSM), 2012 28th IEEE International Conference on*, pp. 592–595, IEEE, 2012.
- [14] W. Mo, B. Shen, Y. Chen, and J. Zhu, “Tbil: A tagging-based approach to identity linkage across software communities,” in *Asia-Pacific Software Engineering Conference*, pp. 56–63, IEEE, 2015.
- [15] A. A. Abbasi and M. Younis, “A survey on clustering algorithms for wireless sensor networks,” *Computer communications*, vol. 30, no. 14, pp. 2826–2841, 2007.
- [16] W. Han, L. Jiang, T. Lu, and X. Zhang, “Comparison of machine learning algorithms for software project time prediction,” *International Journal of Multimedia and Ubiquitous Engineering*, vol. 10, no. 9, pp. 1–8, 2015.

Context and Trust Aware Workflow Oriented Access Framework

¹Tapalina Bhattasali, ²Nabendu Chaki, ³Rituparna Chaki

^{1,2}Department of Computer Science and Engineering

³A. K. Chaudhury School of IT

University of Calcutta, Kolkata, India

¹tapolinab@gmail.com, ²nabendu@ieee.org,

³rchaki@ieee.org

Khalid Saeed

Faculty of Computer Science

Bialystok University of Technology

Bialystok, Poland

khalids@wp.pl

Abstract— Service oriented computing (SoC) changes the way of conducting business as these services are often available on a network. As traditional access control approach may not work in the changed environment, protecting business resource from misuse is a big challenge. Again, static allocation of access right to users will not be an efficient solution as SoC environment changes with time. This paper focuses on design of dynamic access control approach for business process. Here, we propose a context and trust aware workflow oriented access framework. Proposed approach focuses on inter-component relationship where phases are executed either in online or offline mode to avoid performance bottleneck. The concept of static binding in role based access model is extended to support dynamic access control by including context awareness and trust relationship between owner and user. Trust is either directly or indirectly dependent on service level agreement (SLA) compliance, quality of service, reputation and provenance (historical data). In this paper, the framework is designed by mapping proposed access model to workflow instances at run-time. It is validated by workflow net model, where workflow instance can be successfully executed without any interruption and can satisfy soundness property while incorporating proposed access control approach.

Keywords- *dynamic access; workflow; context; SLA; trust; quality of service; service oriented computing*

I. INTRODUCTION

“Any” paradigm supported by ubiquitous computing allows anyone to access objects at any time and from anywhere. This type of access should be carefully controlled to meet security and privacy requirements in dynamically changing service oriented computing environment. One of the key challenges in this type of environment is to design of effective access control approach. Designed model needs to protect resource and service against unauthorized access according to a security policy. It verifies whether a subject to object binding is allowed to carry out or not at run time based on several application specific criterions. In SoC, workflow is used to realize an automated business process by representing execution of multiple relevant tasks in an organized way to meet business objectives. However, growing complexities of business processes demand more flexibility in workflows. Flexibility can be achieved by adopting changes around the environment without changing its originality. This concept can be implemented by considering a framework, which can realize dynamic mapping between authorized users, roles, services and permissions during execution of workflow instances.

This work aims to maintain flexibility as well as security in service oriented computing applications, where anything is considered as a standard service. Services are invoked by huge number of subjects having temporary role and authorization procedures need to be passed through several security domains. Here static binding between subject and object is changed to dynamic binding by means of context and trust aware workflows. Workflow instances are defined and validated by Petri nets. Coverability property of resulting workflow nets are analyzed to verify successful execution of workflow instance after incorporating proposed access control approach, which in turn also analyzes compliance with the service level agreement. Soundness in workflow net model implies that all the tasks are successfully executed in an order, while considering dynamic mapping of access control approach and there are no major flaws in the specification of proposed access framework to interrupt the basic flow of tasks. Valid users can access service at right time under several constraints to realize that flexible operations are performed and flow of tasks goes on. As a result, execution of the application-specific workflow gets more secured, as probabilities for unauthorized data access and data misuses are reduced.

The major contribution of this paper is to propose a theoretical approach that integrates inter-dependent information either in off-line or in online mode to validate successful execution of workflow instance in service oriented framework. The rest of the paper is organized as follows. Section II presents few related works on this domain. Section III identifies scope of the work. Section IV introduces proposed access framework in service oriented computing environment. In section V, the proposed framework is defined as a workflow net model and it is validated using WoPeD simulator. The paper ends with concluding remarks in section VI.

II. RELATED WORKS

Research on access control approach designed for heterogeneous service-oriented computing environment is becoming a very interesting topic in the field of security. There are several access control approaches; each of which has its own merits and demerits. Due to page limitation few popular approaches are mentioned here. Most commonly used access model are - role based access model (RBAC) [1] and attribute based access (ABAC) model [1]. However major drawback of these types of access models is objects are assigned to subjects in a static manner that cannot be changed

with the change of environment. Assigning access rights dynamically can be achieved by context awareness. Therefore, concept of RBAC and ABAC are extended [2, 3, 4] to context aware RBAC and context aware ABAC respectively. In spatial RBAC [5], access to service is granted based on the location and assigned roles of users. In rule based RBAC (RB-RBAC) [6], users are dynamically assigned based on security policy. In situation aware RBAC (SA-RBAC) [7] users are granted to roles and roles are granted to permissions based on the situation information. In trust aware access control mechanism [8] for IoT (TACIoT), lightweight authorization mechanism and a novel trust model are specially devised for IoT environment. In [9], focus is given on synchronization of the workflow and the authorization flow. This is done by assigning time slots to the tasks in the workflow. After analyzing few recent works on this domain, it can be said that there still exist a research gap to balance between flexibility and security.

III. SCOPE OF WORK

Motivated by several identified research challenges [10], major problem can be defined as: How to dynamically bind subject to object according to inter-dependent information considered for a workflow instance that satisfies soundness property of workflow net? In order to solve this issue, a context and trust aware workflow oriented access control approach is considered here as scope of work. It includes design of a dynamic access framework for service oriented computing-

- To bind access right to a subject and object pair for a workflow instance according to change in context and evaluated trust value depending on service level agreement and quality of service.
- To validate access framework with workflow net simulator to ensure that workflow instance goes on smoothly and consistency is preserved in access control approach, if and only if it can satisfy workflow net properties (like well-structured, sound, live, bounded, and deadlock-free).

IV. PROPOSED APPROACH

This section introduces proposed context and trust aware workflow oriented access framework that depends on several components. Proposed approach is novel in the sense that context-aware dynamic access approach based on inter-component relationship maps with workflow instance and is defined by simple workflow net model. Thus, it can be easily implemented for a real life scenario. Besides, it can capture status of workflow instance for deriving successful execution of proposed access model to ensure safety and validation of the approach. It includes design of a dynamic access framework based on context information and evaluated trust value on workflow management.

A. Context and Trust Aware Access Approach

It integrates the concept of workflow based access control, context aware role based access control, trust based access approach control, and SLA aware trust model. Proposed access

approach can be defined as a seven phase and nine tuple access model. Among the seven phases, phase 1 is only considered for off-line mode processing to avoid performance bottleneck. This approach is designed to meet the basic requirements of access control in service oriented computing framework. It supports fine granularity, scalability, context-awareness and capable to adapt run time changes. Figure 1 represents proposed access control approach.

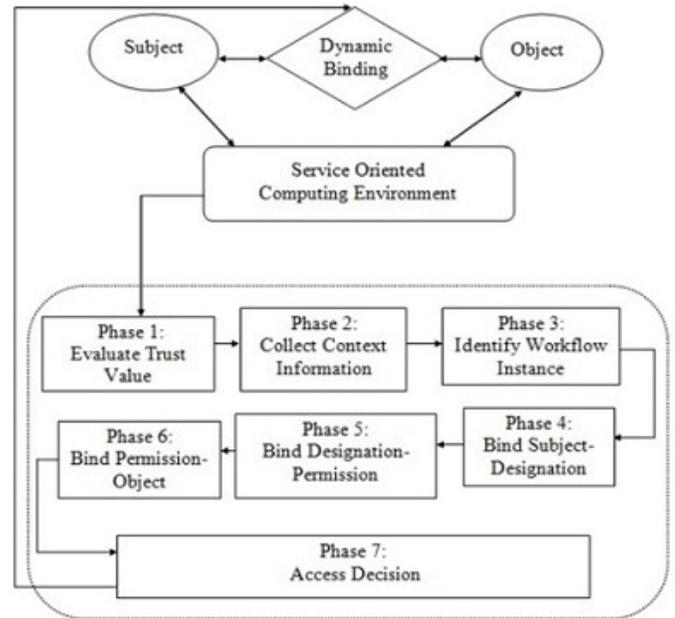


Figure 1. Context and Trust Aware Access Control Approach

Figure 2 represents basic building block of proposed access model.

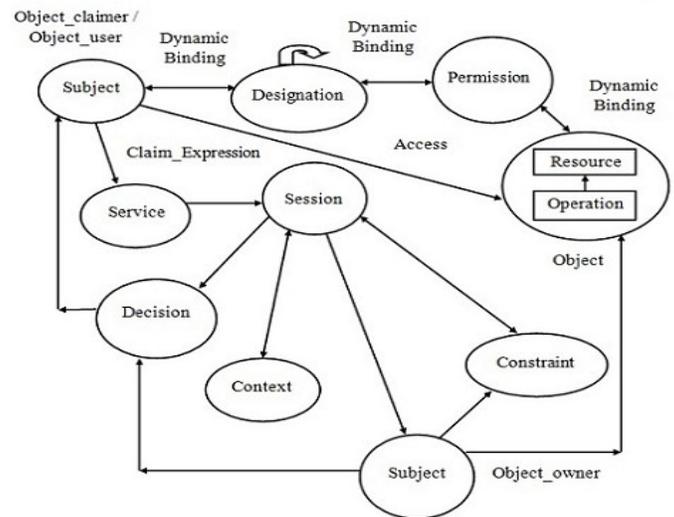


Figure 2. Building Block of Proposed Access Model

Proposed Access Model = {S, O, D, Per, Oper, Auth, Po, E, WF(t)}, where S → subject, O → object, D → designation, Per → permission, GL → granularity level, Auth → authorization decision, Po → policy, Oper → operations, E → expression, WF(t) → workflow instance. The major elements of proposed access model are discussed below.

S: Entity that takes action on an object. It is categorized into object_owner, object_claimer, object_user, relevant_entity.

O: Entity that is acted upon by a subject. It can be either resource or service. They are protected by set of policies.

D: It represents subject's functional role at a specific instance. There are several classes of D. Intra-class D can be easily interchanged for a workflow instance and interchange of inter-class D is based on dependency between two classes. Exchange of D within intra-class and inter-class is defined by designation partition rule, where two designations cannot be assigned to same subject within a session of workflow instance.

Per: This is the approval to perform certain operations on objects by the subjects. This is generated by access-right generator module. Per set includes {read, write, execute, modify, activate, deactivate, allow, disallow, upload, download}.

Oper: Per and Oper are related to each other. Based on Per, Oper is executed on object by subject at a time.

Auth: Decision of authorization is generated to enable dynamic binding at run time.

WF(t): It represents executable workflow vector at session time t with the lifetime T_1 to specify workflow instance.

Context(c): It is related to each entity involved in a workflow instance of the application. Context generally depends on {Time, Location, Environmental parameters,...} during its execution time.

Constraint (cons): It is expressed as a regular expression cons: = union (clause₁,, clause_n), where clause:= intersect(cond₁,...,cond_m), and cond:= <c> <op> <context_val> AND <trust_val>, where op is a logical operator from set { <, >, <=, >=, ≠ }.

Po: Set of rules is applied to model to get service as expected. It can be represented as a pentuple. Po: = < S, D, O, Oper, cons >.

E: These are used to express either claim_request or inter-dependent information in order to specify binding between subject-designation, designation-permission and permission-object.

Object_Access (OA): It can be represented as a quadruple. OA:= <S, Oper, O, dc>, where dc implies dynamic context considering every context within context set. OA is granted if and only if all tuples of it evaluate true under dc.

Hierarchical_Designation (HD): Designation is considered in a hierarchical manner to apply the concept of inheritance. If permission is assigned to a lower class of designation, then it is also assigned to all the higher class designations.

Object_Hierarchy (OH): It supports a subject to access different granularity levels of objects in a hierarchical manner. If a subject has the right to access object with the highest granularity level, then it has also right to access the lower granularity levels of that subject.

Dynamic_Binding (DB): It is a many to many mapping between subject and object at run time. It is mainly of three types- subject to designation mapping, designation to permission mapping and permission to object mapping.

Access_Dependency (AD): It represents the relation between access decisions in workflow instances. Such as,

- AD2 can be activated only after AD1 is finished.
- AD1 and AD2 must be mutually exclusive for a workflow instance.
- Permissions granted to AD1 can be delegated to AD2, when AD1 is aborted.
- AD1 and AD2 must be granted to two different subjects.
- AD2 can be activated only after AD1 is defeated.

Figure 3 represents inter-dependency among multiple components of proposed approach.

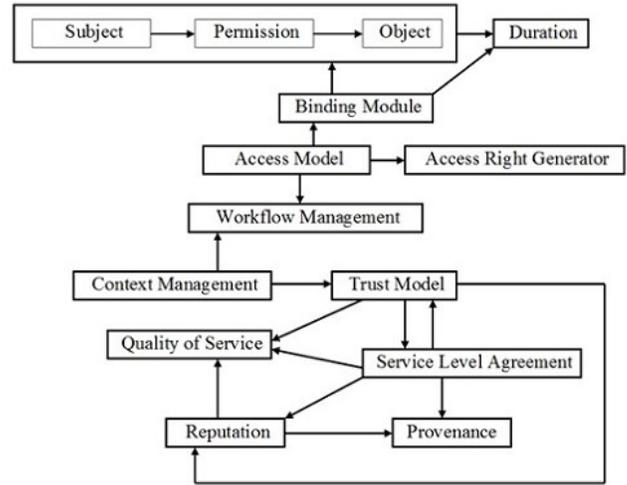


Figure 3. Inter-Component Relationship of Access

Inter-component relationship of access model is identified below. Left hand sides represent determinants and right hand sides represent dependents.

- Provenance → {Reputation, Service Level Agreement}
- Reputation → {Service Level Agreement, Trust Model}
- Service Level Agreement → Trust Model
- Quality of Service → {Reputation, Service Level Agreement, Trust Model}
- Trust Model → {Service Level Agreement, Context Management}
- Workflow Management → {Context Management, Access Model}
- Access Right Generator → Access Model
- Binding Module (operation, expression, designation) → Access Model
- {Subject, Permission, Object} → Binding Module (operation, expression, designation)
- Duration → {Binding Module, Subject, Permission, Object}

B. Integration Logic based on Inter-Component Relationship

Integration logic of inter-component relationship works in two modes- off-line and online mode to avoid performance bottle-neck that may arise if components are integrated during dynamic mapping at online mode. Procedural logic of proposed access approach is presented below for both off-line and on-line mode.

Off-line mode:**begin**

1. construct SLA between subject S_m and S_n
2. set $trust_set := \{1:1\ trust, third_party\ trust, derived\ trust\}$
3. select $trust_type$
4. construct soft trust vector for each subject pair
// soft trust is trust without hard copy proof
5. initially set provenance to NULL
6. check QoS parameters
7. compute reputation from (provenance, QoS) value
8. check SLA_compliance (provenance, reputation, QoS, trust)
9. evaluate trust (availability, degree of separation, regulatory compliance, recovery probability, successful access)
10. construct SLA aware trust
11. evaluate inter_component trust (SLA_compliance, reputation, QoS) for $trust_set$
12. $trust_val = union(trust, inter_component\ trust)$
13. store $trust_val$ //to be used in online mode
14. update provenance value, reputation value, $trust_val$
15. evaluate trusted_relation vector (provenance, reputation, QoS, SLA_compliance, $trust_val$, timestamp)
16. construct trust_aware access model
17. generate access policy from access right generator

end**Online mode:****begin**

1. construct workflow vector $WF := \{t_1, t_2, t_3, \dots, t_n\}$
2. S_i initiates service request SR // $S_i \rightarrow$ subject
3. start session
4. analyze SR
5. identify workflow instance (WFI) for subject S_i
6. set $WFI := \{subject, task\ set, session\ time\}$
7. collect context_info
8. identify $trust_type$
9. check $trust_val$
10. if $trust_val = +1$ then
11. S_i is honest
12. else if $trust_val = 0$ then
13. S_i is honest but curious
14. apply stronger Po // stronger policy
15. else if $trust_val = -1$ then
16. S_i is dishonest
17. block S_i
18. construct context_vector := {context_info, trusted_relation}
19. update WFI based on context_vector
20. update access model
21. identify status (subject, designation, attributes) for WFI
22. identify expression and operation
23. set session_validity to duration
24. map WFI to access_model
25. bind (subject, designation, attributes, permission, object, operation, duration, context_vector)
26. dynamic_binding{(subject, designation), (designation, permission), (permission, object)}
27. if session_validity expires then
28. check WFI status
29. if WFI status= "finish" then
30. grant access
31. successful binding
32. WFI executes without interruption
33. else
34. deny access

35. binding not successful

36. block WFI

end**C. Mapping of Access Model to Workflow Instance**

Workflow management module is mainly used to design workflow model to improve the efficiency of business process. Workflow instances are executed either in serial order or in parallel or in iterative way. It is based on a task set, where each task has a life cycle having states like ready, active, blocked, finished, and invalid. A mapping needs to be considered between workflow instance and access approach to determine the effect of workflow on access control. Figure 4 represents context and trust aware workflow oriented access approach.

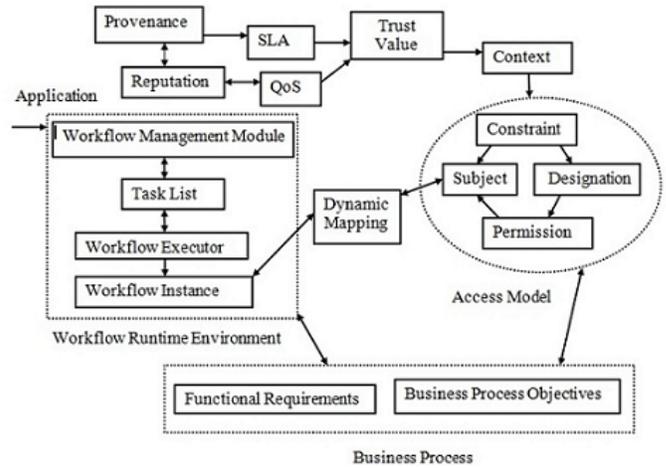


Figure 4. Context and Trust Aware Workflow Oriented Access Approach

Access dependency decides the control flow of workflow instance. As permissions are granted according to the current status of workflow instance, access control matrix (ACM) is extended by one dimension.

Therefore, $ACM_{dyn} : S \times O \times TWFI \rightarrow P$ (per)

where $S \rightarrow$ subject, $O \rightarrow$ object, $TWFI \rightarrow$ task set for a workflow instance and P (per) \rightarrow power set of all permissions.

It is assumed that subject S_i requests a specific permission Per_j for object O_k . ACM_{dyn} is used for this purpose to check current status of WFI for all active tasks.

- If Per_j entry is included within ACM_{dyn} , it is granted. If task t is not activated any more, then Per_j is revoked again.

- If Per_j entry is not included within ACM_{dyn} , then Per_j is not granted. The extension from two to three dimensions increases the number of entries in the matrix.

The number of entries in $ACM_{dyn} := |S| \times |O| \times |TWFI|$

All permissions from ACM are also included in ACM_{dyn} . However, it can be said that ACM_{dyn} grants access more restrictively than ACM. The risk of data misuse in ACM_{dyn} is lower than in ACM. Therefore, it can be said that workflow oriented dynamic access control framework is more secure than others access control model.

V. ANALYSIS OF ACCESS FRAMEWORK

Workflow net is used here to represent access framework. The workflow specification represents a template, which is used to generate workflow instances by workflow executor. Order of execution of activities may be different at different instances. Workflow net model is defined and analyzed here by WF-net simulator WoPeD [11]. It has the following characteristics. Tasks in workflows correspond to transitions in workflow net. Executing a task corresponds to the firing of a transition. Marking of a workflow net represents the current state of a workflow. The control tokens represent the state of the workflow. The flow relation indicates how the tokens can move in the net. The control flow is determined by the flow relation and the start marking. The execution of workflows is supervised by the workflow executor and workflow management module to create and manage various instances of a workflow. Different sequences are created for every workflow instance. Access control is enforced upon the objects of the workflow instances.

In order to implement the access control mechanism, workflow management module generates and controls a workflow net for every instance. Every time a transition fires in an instance net, the marking has to be changed. Although we have examined proposed access framework at different scenarios to validate it at different workflow instances, we cannot present here the details of analysis due to limited space. It is sufficient to verify only a single instance to prove workflow net properties. We, therefore, present here only one instance. It needs to be determined whether a workflow net is completely executable or not, i.e., if the end marking is reachable from the start marking. This is analyzed by token game analysis and coverability graph. To check whether workflow specification is syntactically and semantically correct, concepts like liveness, deadlocks, and soundness are considered. Smooth execution of workflow instance is determined if source token reaches to terminal place successfully and all places are covered by the graph. This can also ensure consistency in proposed approach.

TABLE I. PLACES AND TRANSITIONS OF WORKFLOW NET MODEL

Place		Transition	
p1	initiate (with token)	t1	object_claimer
p2	service_request		
p5	workflow executor	t2	session
p7	context		
p8	trust	t4	WFI (workflow instance)
p11	constraint		
p12	granted		
p13	denied	t14	access_decision
p14	S-D binding (subject-designation)		
p15	D-P binding (designation-permission)	t16	session_end
p16	P-O binding (permission-designation)		
p19	terminate		

Table I represents places and transitions of workflow net model presented in figure 5. Qualitative analysis of this model by WoPeD tool shows it can satisfy workflow properties. It is

well structured and can satisfy structural feasibility. This model is sound that can satisfy workflow net property, initial marking, boundedness and liveness. There are no wrongly marked places in the net models. As there are no unbounded place, boundedness properties are satisfied. As number of token in a place is limited to 1, it is bounded. It is also considered as a property of safeness. Conservation properties are also satisfied here as number of token remains 1 (constant) before and after execution. Liveness property checks the probability of deadlock. As there are no non-live or dead transitions, it can be said that the models satisfy liveness property too. Therefore the designed net model is considered as sound and safe. There are no probabilities of deadlock.

TABLE II. PHASE WISE FLOW-IN AND FLOW-OUT

Transition	Flow-In	Flow-Out	Phase
t1	p1	p2, p8	1
t2	p2, p8	p5, p11, p7	2, 3
t4	p5, p11, p7	p14, p15, p16	4, 5, 6
t14	p14, p15, p16	p12, p13	7
t16	p12, p13	p19	7

Table II represents flow-in and flow-out places associated with each transition of workflow instance presented in figure 5.

TABLE III. MARKING OF WORKFLOW NET COVERABILITY

From Marking	To Marking	Transition
1000000000000	0100100000000	t1
0100100000000	0011000100000	t2
0011000100000	000000010110	t4
000000010110	0000010000000	t14
0000010000000	000000001000	t14
0000010000000	0000001000000	t14
0000000001000	0000000000001	t16
0000001000000	0000000000001	t16

Table III represents marking of workflow net coverability to show that all the places and transitions of the model can be covered and no two vertices have same marking. Therefore, it can be said that workflow net model is completely executable. The execution sequence of workflow instance in figure 5 is as follows.

sequence 1: {p1} → {t1} → {p2, p8} → {t2} → {p5, p7, p11} → {t4} → {p14, p15, p16} → {t14} → {p12} → {t16} → {p19}.

sequence 2: {p1} → {t1} → {p2, p8} → {t2} → {p5, p7, p11} → {t4} → {p14, p15, p16} → {t14} → {p13} → {t16} → {p19}.

There are 5 transitions {t1, t2, t4, t14, t16} and 12 places {p1, p2, p5, p7, p8, p11, p12, p13, p14, p15, p16, p19}. In t14, AND-join-XOR split is used to combine (AND) the effect of the activities in S-D binding, D-P binding and P-O binding and then split (XOR) the flow either in p12 (granted) or in p13 (denied). In t16, XOR-join is used to either consider effect of p12 or p13 before termination.

Soundness property of workflow net model of proposed access control approach ensures that - there are no dead transitions, every transition state is reachable from source place to sink place triggered by a firing sequence, sink is the only place reachable from source place with at least one token in sink place.

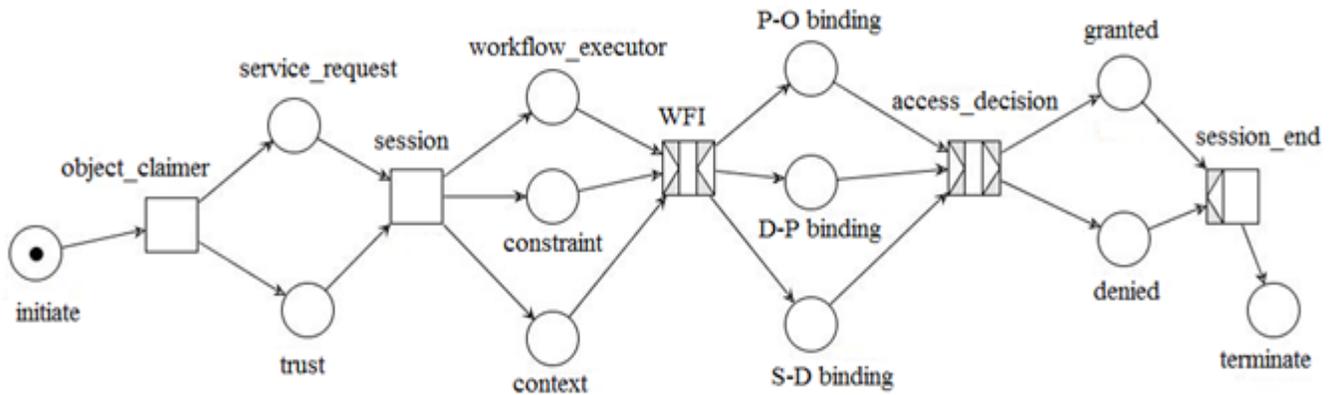


Figure 5. Validation of Context and Trust Aware Workflow Oriented Access Framework by WoPeD Simulator

VI. CONCLUSION

A major challenge in service-oriented computing environment is the design of effective access control approach. In this paper, a context and trust aware workflow oriented access framework is presented to solve challenges associated with business processes in Service Oriented Computing. General RBAC model cannot solve large number of temporary users. Therefore, RBAC model is not suitable for access control of SoC. Proposed approach is a flexible access control approach that binds subject to object in three phases- S-D binding, D-P binding and P-O binding based on inter-component relationship. Access model is mapped to workflow instance at run time. Validation of proposed framework is achieved by simulating through WoPeD tool. Soundness property of workflow net model guarantees the absence of livelocks, deadlocks, and other anomalies. After analyzing successful execution of various workflow instances as discussed in section V, it is established that there is no inconsistency and incorrectness in proposed dynamic access framework. It satisfies smooth execution of workflow instance without any interruption during analysis of model with token game approach.

Proposed approach is compared with workflow-oriented attributed based access control approach designed for SoC [12] to prove effectiveness of our approach. The work is going on to evaluate the performance of the proposed framework in healthcare domain, which cannot be presented here due to shortage of space. In the future, we will deploy this model and prove its validity.

VII. ACKNOWLEDGEMENT

This work is supported by Information Technology Research Academy (ITRA), Government of India under ITRA-Mobile grant (Ref. No. ITRA/ 15(59) / Mobile / RemoteHealth / 06).

VIII. REFERENCES

- [1] Z. He, L. Wu, H. Li, H. Lai, Z. Hong, "Semantics-based access control approach", *Journal Of Computers*, Vol. 6, No. 6, pp. 1152-1161, 2011.
- [2] R. J. Hulsebosch, A. H. Salden, M. S. Bargh, P. W. G. Ebben, J. Reitsma, "Context sensitive access control", In proceedings of SACMAT, pp. 111-119, 2005.
- [3] J. Zheng, K. Q. Zhang, W. S. Zheng, and A. Y. Tan, "Dynamic role-based access control model", *JSW*, 6(6):1096-1102, 2011.
- [4] J. Lee, M. Winslett, J. Basney, and V. Welch., "The trust authorization service", *ACM Transactions on Information System and Security*, VOL.11(1), 2008.
- [5] H. Zhang, Y. He, Z. Shi, "Spatial context in role-based access control", In proceedings of ICISC, pp. 166-178, 2006.
- [6] B. Carminati, E. Ferrari, A. Perego, "Rule-based access control for social networks", In proceedings of OTM Workshops, Springer, 2006.
- [7] S. S. Yau, J. Liu, "A situation-aware access control based privacy-preserving service matchmaking approach for service-oriented architecture", In proceedings of ICWS, pp.1056-1063, 2007.
- [8] J. B. Bernabe, J. L.H. Ramos, A. F. S. Gomez, "TACIoT: multidimensional trust-aware access control system for the Internet of Things", *Soft Computing Journal*, Springer, pp. 1-17, 2015.
- [9] V. Atluri, W.K. Huang, "An Authorization Model for Workflows", In Proceedings of European Symposium on Research in Computer Security, Springer, 1996.
- [10] J. Zhu, Y. Zhou, W. Tong, "Access Control on the Composition of Web Services", Proceedings of the International Conference on Next Generation Web Services Practices (NWeSP), 2006.
- [11] Workflow Petri net Designer, Available online at: <http://woped.dhbw-karlsruhe.de/woped>
- [12] G. Zhanga, J. Liub, "The Study of Access Control for Service-Oriented Computing in Internet of Things", *International Journal of Wireless and Microwave Technologies*, vol. 3, pp. 62-68, 2012. Available online at <http://www.mecs-press.net/ijwmt>

Integrated Data Management System for Data Center

Debasis Dash¹, Juthika Dash², Younghee Park^{3*}, Jerry Gao⁴

Computer Engineering Department
San Jose State University
California, USA

{debasis.dash¹, juthika.dash², younghee.park³, jerry.gao⁴}@sjsu.edu

Abstract— Cloud computing has become the most popular computing architecture in the world. The multiplicity of available platforms for end users makes it very inconvenient for service providers to manage, calling for a unified approach for resource management and security in virtualized data centers in order to achieve seamless cloud management. In this paper, we present data center management as a service (DMaaS) as such a unified approach, using multi-tenant software. Virtualized data centers pool cloud resources used to meet enterprise needs, mostly related to storage and processing. The proposed work provides a service model for the cloud infrastructure and supports multiple tenants to connect their data center management in a shared and secure manner. Our service manages disparate virtual environments using a single interface.

Keywords- Datacenter, OpenStack, VMWare hypervisor, server virtualization, cloud, SaaS, analytics, recommendation, access control list

I. INTRODUCTION

Major challenges arise in providing security and resource management in virtualized data centers, which pool computing and networking resources. Some papers have explained issues and benefits involving the use of virtualization in data security services in cloud computing [6] [7]. Some of the security issues faced while managing virtual infrastructure include maintenance, configuration, and updates for software, and generally maintaining consistency and integrity [6]. High performance and failure recovery in data centers are essential since so many users depend on the data center computing power. The complicated underlying services and systems in data centers create challenges for heterogeneous data center infrastructure. It is hard to manage system resources and monitor cloud computing in such diverse data centers.

The challenges of monitoring data center performance [1], which is itself a synergy of server performance, application performance, and security threats, multiply when disparate virtualization platforms and monitoring tools are used. In addition, an on-demand, mandatory security monitoring system in the virtualized datacenter environment is required, one that won't depend on pre-installed guest components. The ultimate aim is to utilize, deploy, work, and monitor the security in a large-scale virtualized datacenter cloud environment, such as the IaaS cloud [1]. The need for virtualization administration and automation has been emphasized [3], requiring a unified

approach to data center management and security monitoring. Our proposal offers a unified solution to manage, integrate, and monitor a heterogeneous datacenter.

In this paper, we propose a data management system called DMaaS (Data Management as a service) in order to integrate different data center environments into a single platform for easy configuration and efficient management. The proposed system is composed of different service layers and critical subsystems to provide a single platform to manage different data centers depending on user requirements. The proposed solution provides an abstraction interface to manage data centers with disparate resources in a virtualized infrastructure. Users can use a single platform to monitor and manage their data centers with a high degree of usability. The proposed system offers resource recovery and management, user pattern recognition, and recommendation systems to provide best-effort services for end users. Through monitoring system resources and user activities, our system provides a wide range of metrics for VM's in the data center. Our experimental results demonstrate that our system can neutralize overhead in a single data center, and efficiently assign system resources to different data centers by using various profiling techniques in terms of systems and users.

Section II discusses our motivation for engaging in this work. Section III presents the proposed system with a detailed description of system architecture. Section IV shows experimental results with our implementation testbed. Section V discusses directions for future work. Section VI explains related work. Lastly, our conclusions are in Section VII.

II. MOTIVATION

Virtualization has led to a dramatic rise in the development of data center technology, especially in cloud platforms. The modern cloud environment offers different hardware configurations designed for classes of applications or user requirements. A typical cloud environment supports multiple generations of hardware simultaneously. Heterogeneous advanced technologies make it difficult to manage various data center services and to efficiently utilize data center resources. Therefore, data center management software must be independent of any platform. An integrated service management system to unify heterogeneous data centers is needed to minimize operation costs and provide high usability.

Figures 1 and 2 show two cases in common use to illustrate the motivation behind our research. Figure 1 shows two companies, A and B, which have their own virtualized environments, each in their own data center. If company A integrates company B's data center, with its different

*This author is a corresponding author.

DOI reference number: 10.18293/SEKE2016-128

infrastructure, company A must first update its existing data center software to create a unified platform that can seamlessly manage both data centers without changing the underlying infrastructure. For such an integrated single data center working with disparate virtual infrastructures, it is critical to have a common monitoring and managing platform.

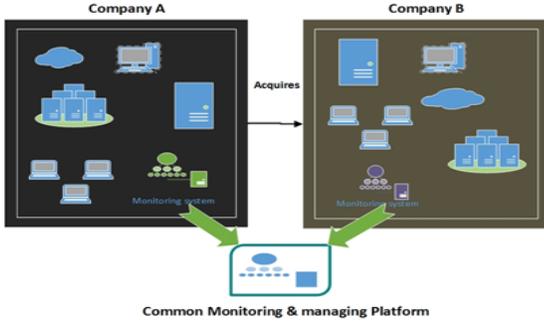


Figure 1. Different virtual environments for two companies

Figure 2 illustrates another case. A company has multiple data centers in different geographical locations with different virtual environments. Its multiple data centers have different capabilities and different sets of management environments. Managing all the data centers in a unified manner is challenging, but necessary, in order to provide efficient resource management and to avoid security problems. Therefore, this paper proposes a unified management system to manage and control multiple data centers from a single piece of software and a single location.

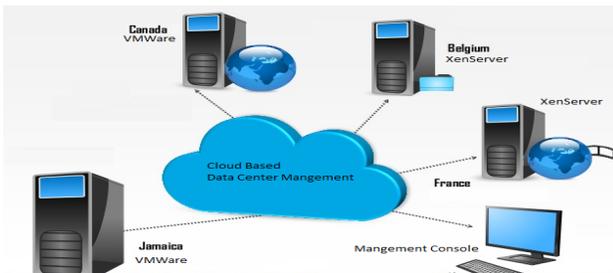


Figure 2. A company has multiple datacenters in different geographical locations with different IT environments.

III. SYSTEM ARCHITECTURE

A. Overview

We propose datacenter management as a service (DMaaS) for streamlined IT operations at heterogeneous data centers. The proposed solution is able to process a large amount of data in a virtual system that monitors resource usage and threats in the data center. The solution provides a unified platform implemented in Multi-tenant Service Oriented Architecture, which means that a group of users share common access to every tenant including every tenant’s data, configuration, user management, and individual tenant functionalities.

Multi-Tenant service oriented architecture provides a cost effective highly scalable software model. This architecture reduces the user’s initial capital expenditure (capex) by sharing the cost of infrastructure and maintenance. Service oriented architecture (SOA) helps the model to scale easily as each functional module is loosely coupled with the others. The distributed nature of SOA provides high availability to the system. The proposed solution utilizes this architecture to achieve a distributed, scalable, cost-effective software model.

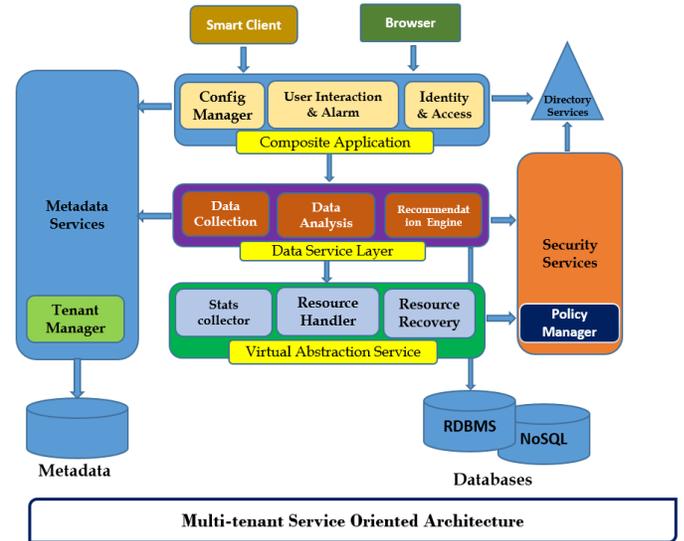


Figure 3. System Architecture for DMaaS (Datacenter Management as a Service)

Figure 3 presents the system architecture we propose to achieve our goal, our Datacenter Management as a Service (DMaaS). The proposed system consists of different service layers: metadata, analytics, directory, and security services, with frontend user interfaces including a smart client and a browser. To support these complex services, there are three main system components: a presentation system, a business process service system, and a data analysis system. The proposed solution comes under the SaaS (Software as a Service) model. We have designed our model to utilize this service-oriented architecture in a multi-tenant environment. The following sections describe each individual component in detail.

B. Various Service Layers

This proposed system architecture has various service layers: metadata, analytics, directory, and security services, in order to integrate heterogeneous data centers into a single unified platform to support streamlined data center management.

The *Metadata Services* layer is provided by the tenant manager, a major element to ensure sanity among users from different enterprises. This provides a primary means of customizing and configuring a new unified cloud environment for each user while collecting users’ requirements from different infrastructures. Based on user input, our system

provides a service to monitor parameters installed by end users. It supplies a user-friendly control environment enabling users to configure the service and to monitor workflow and business rules.

The **Analytics Services** layer performs data cleansing and analysis based on Spark, an open source cluster computing framework. It interacts with the business process service system and data storage to keep track of system resource usage with various large data from systems and networks. Through data analysis, the proposed system can provide a single unified data center for each user by using the system and network resources efficiently through the results obtained from data analysis.

The **Directory Services** layer provides access control functionality to the service based on role-based access control. In other words, system requirements are affected by different job privileges for each user. Each user has a profile based on roles and responsibilities for configuration management and based on web service recommendations. This provides user authentication and authorization based on different user profiles.

The **Security Services** layer monitor users' activities for system and network resources. Through user profiles and usage patterns, it provides a security monitor to check any malicious abnormal behavior in the allocated data center. It monitors system activities through memory usage, CPU usage, file access patterns and access permission, and network usage with protocols and port numbers. Through this security services layer, our system can block malicious users and replace compromised configurations into a clean state environment by configuring a new data center on the fly. It can be achieved since all the data for each user is saved in data storage.

The front-end user interfaces (i.e. smart client and browser) utilize these different service layers to visualize resource usage, anomalies, and statistics provided from the data center. The smart client is a mobile application and the browser is a web client to connect into the data center management service in a secured manner.

C. Three Main Systems

To support these integrated services, there are three main system components: a presentation system, a business process service system, and a data analysis system. First, the presentation system provides access to service configuration, alarms, and recommendations using modern web services. It is integrated to provide data visualization in a most effective manner. Second, the business process service system provides the functionality to connect to the datacenter to collect system logs, network log, and interaction with hypervisors for collecting usage metrics. The recommendation engine is a part of this system and interacts with the analytics service layer to query required data and provide recommendations using these data. Lastly, the data analysis system analyzes the system log, resource usage patterns and user activities on Spark to provide better recommendations to end users while maintaining efficient data management.

These three subsystems have important components to achieve DMaaS: resource discovery, resource management, role-based access control, usage pattern recognition, and a recommendation system as described below.

Resource discovery supports scale-out of the data center. Application modules work as a controller with a global view of resource availability and usage. When a user requests computing resources, if the local datacenter doesn't have sufficient resources then it can acquire resources from a different datacenter under the same enterprise. It will use overlay technology to stitch both virtual systems. Hence, users can leverage the availability of remote resources.

Resource management plays an important part by providing a hyper-call abstraction to the application service. Virtual Abstraction Interface (VAI) implements hypervisor-specific implementation for translating an application request to the appropriate hyper call. Major functionality involves resource creation, maintenance, and extraction of resource usage patterns. VAI sends all the requested data using a northbound interface towards the application module.

Role-based access control provides the access control list for role-based usage. The system implements various user levels and assigns access control based on roles. This provides an additional level of security to the hosted service. There are three user roles: high-class, medium-class and low-class. Users who compute intensive scientific flow analysis and computing programming belong to the high class that requires high memory and CPU resources. However, users having common jobs working on documents, checking emails, and surfing can be classified into the low class. These users usually need smaller resources compared to high-class users. Others are classified into medium-class users. This class system is based on user profiles from past or current activities through user pattern recognition.

User Pattern Recognition is useful for efficient usage of resources available in the resource pool. It goes hand in hand with the recommendation system of DMaaS described in the next section. It can detect abnormal users if any resources are being under or over-used by a particular user. This information is given to the recommendation system, which can provide recommendations for better resource usage for increased performance standards. This module is useful in tracking user activities, and analyzing resource usage. The information is also useful in training the recommendation system to provide accurate and useful recommendations.

The **Recommendation System** for DMaaS has been designed to improve the user experience. Two main features of the recommendation system are providing recommendations for better resource usage and providing recommendations for a new user to ease the creation of Virtual Machines (VM) depending on his/her preferences (processing, memory, network bandwidth need). It generates recommendations for a user depending on the user's usage of VM depending on user profiles. For example, some users utilize VM for surfing the web in the low class, or other users need to have VM for intensive computing in the high class. These extreme cases

must each be handled differently by our proposed system for efficient resource management and better service in the data center. To achieve our goal for best-effort services, we record user usage in the data center as a training model. Based on training data, the proposed system provides a recommendation to match user preferences with current system status. When a user input is given, the proposed system makes a recommendation based on the highest similarity between the current input and the profiled usage. The similarity is computed with *Jaccard Index*. We extract a set of features, such as memory usage, CPU usage, duration and frequency to use the data center, job objectives depending purpose to use the data center and geographical locations. To compute the values of Jaccard index, we assign a numerical value from one to ten depending on scale for each feature.

A cold start problem would occur if there were absolutely no users registered for the DMaaS system and hence the recommendation system cannot generate any recommendations. This potential problem will be addressed by asking users generic questions pertaining to their preferences of activities performed on the computer, such as how frequently they carry out computer intensive work. Asking these questions will help in developing their user profiles, which will be used in the future for training the model and generating recommendations.

Finally, our proposed solution provides many visual charts for easy understanding of resource usage patterns. The web service includes various graphical interfaces to create and manage virtual machines, data centers, adding new resources, logging, utilization, and recommendations.

IV. IMPLEMENTATION AND EXPERIMENTS

A. Implementation

The test setup is comprised of two systems with a quad-core processor with 16GB of RAM. One of the systems is installed with VMWare ESXi and other system is installed with OpenStack software. These two systems are used to simulate individual data centers. Two cloud instances are required to run web service and data analysis for the recommendation system. Proof-of-Concept is implemented in the three following major components: Web Service, Virtual Abstraction Interface (VAI) and Recommendation Engine.

Web service includes an access control list for role-based access to the web service, using LDAP. It receives user requests, processes them, and creates southbound calls. It also visualizes resource utilization, recommendations, and logging. It is implemented on Ruby in Rails, LDAP, high charts and cloud database as a part of the web service development. VAI decouples the hypervisor-specific interface from the application layer. It consumes java-based VMWare. It interacts with API and OpenStack. Message brokers like RabbitMQ are used for inter-process communication. Lastly, we have implemented our recommendation engine using java language. This module consumes data from VAI to train the recommendation engine.

B. Experiments

The system will be tested for accuracy, security, efficiency and performance. The accuracy of the system mainly depends on the recommendations provided, the resource allocation and keeping track of over and underuse of resources. The security feature of the system defines controlled access to the system, i.e. only authorized and registered users can access the system. To implement security features, DMaaS has incorporated the use of Access Control Lists. The system will be evaluated on efficiency and performance. Efficiency can be determined by how quickly the system can adapt to changes and how quickly the system responds to user requests. Handling of failures will also determine the efficiency of DMaaS. Performance can be determined by how efficient the resources are from the resource pool used.

A few experiments have been conducted to evaluate these systems. Currently, each sub-module has been evaluated separately. One such experiment was conducted to check the accuracy and performance of the Recommendation System sub-module. The experiment was to verify whether the system was getting trained appropriately. A few users were added to the system with their preferences about VM usage. Each user request was considered as a training rule and each such training rule was given a score. Whenever users selected the exact same rule and the exact same VM type, the score was increased. A threshold was maintained. When the score of a rule reached the threshold, it was considered that the particular VM type associated with the rule was popular and hence could be recommended to other users with similar preferences. The experiment was successful for a small number of users. More user data will help in increasing the accuracy of this sub-module.

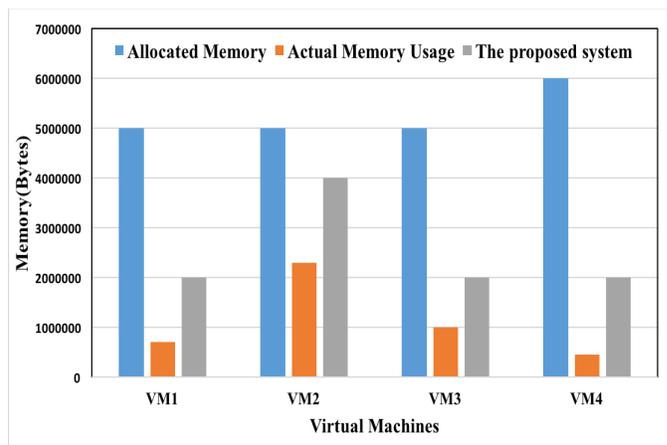


Figure 4 - Memory utilization vs memory allocation

Figure 4 shows the memory utilization for individual virtual machines versus allocated memory inside the data center. As we can see that some virtual machines were underutilized while a few were closer to the maximum allocation. In other words, the allocated memory prior to using our system showed less than a 30% utilization, compared to the allocated memory. However, the proposed system optimizes the resource allocation in terms of memory as shown in Figure 4. The proposed system uses this information to recommend better

resource distribution for current or future users. We can increase more than 70% utilization while reducing the memory size depending on actual memory usage patterns for end users.

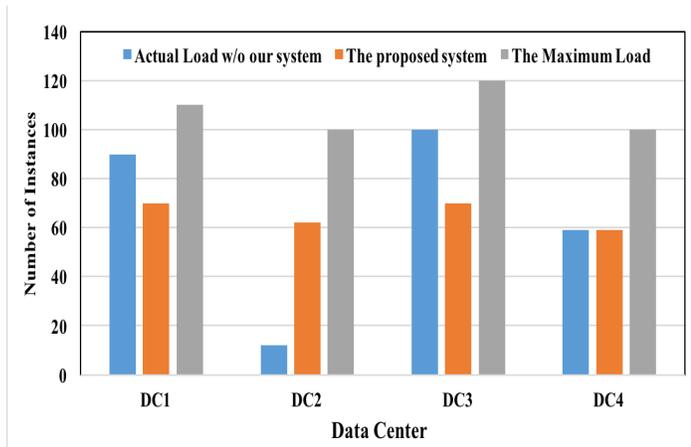


Figure 5 - Datacenter utilization (based on number of VMs)

Figure 5 shows data-center level workload distribution. The chart consists of four data centers identified as DC1 to DC4. The blue bar indicates the current actual load at each data center. The orange bar represents the workload recommended for our proposed system. The gray bar indicates the maximum workload for user requests that each data center can accommodate. The proposed system can recommend current or future users to move some of their workload from the DC1 and DC3 data centers to the DC2 data center to balance the workload across all data centers. Real-time monitoring results can be immediately applied to the current system in the data center for balanced workload distribution.

The proposed system monitors system resource utilization in terms of CPU, memory, disk, IO and network bandwidth utilization. This information enables the proposed system to understand average usage for different users and different virtual machines. Figure 6 depicts memory usage among various virtual machines for a given data center. The mentioned parameter data types are collected from all the virtual machines running in the individual data center and displayed after user login to the system. In addition, it also gave the resource utilization for the various systems.

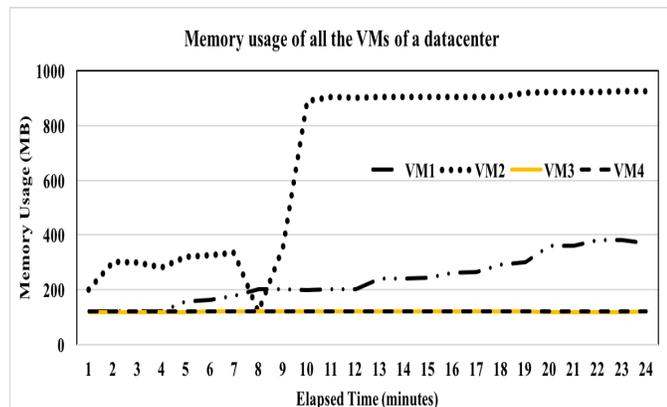


Figure 6. Data center memory usage

V. DISCUSSION AND FUTURE WORK

DMaaS (Data Management as a Service) is a cloud-based service that can be deployed in the public cloud, as illustrated in Figure 7. This figure demonstrates a typical deployment scenario. Two companies, Company A and Company B, utilize our DMaaS platform. The companies are able to create and monitor VMs and get recommendations for them across different hypervisors like VMWare, OpenStack, and Xenserver [12] using our proposed solution. The data center administrators of both these companies find it easy to maintain security, monitor activity, and managing workload using recommendations through DMaaS. Major components required for the deployment are a cloud computing service, load balancer, database, and firewall.

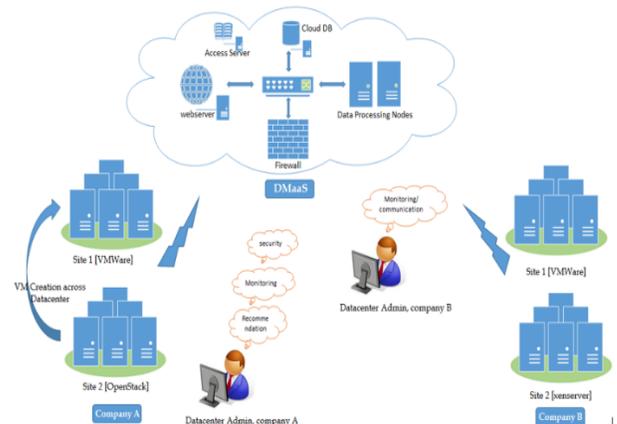


Figure 7. An illustration of DMaaS Deployment

The Cloud computing service uses multiple cloud instances. This includes both small and large cloud instances. All the micro instances will be used for web service hosting, the virtualization layer, and security analysis. The proposed system in this paper utilizes a large instance for the data analysis and recommendation system. The proposed system also requires a load balancer for balanced workload distribution. The load balancer can configure new instances if the traffic rate increases. It provides easy scaling with respect to traffic and resource load.

A firewall can be used along with our security features to add security towards the enterprise data pipeline. This will provide additional security for accessing data service from the enterprise pipeline towards DMaaS service.

In future work, we will work on dynamic adjustment features by using virtualization techniques, such as resizing memory and migrating data, to shed further light on our analysis results and to recommend a better environment for an efficient and streamlined data center management system. In addition, the proposed system will be upgraded through fine-grained recommendation systems by using machine-learning techniques and resource monitoring systems by using logging technologies and computer forensics. Through multiple use case study, we will thoroughly evaluate our proposed system based on benchmarking tools.

VI. RELATED WORKS

A pre-existing monitoring and reporting tool for VMWare VSphere [9] and Microsoft Hyper-V was instrumental in providing a detailed case study of monitoring of virtual machines and environments, notification and alert mechanisms for failure and backup, and troubleshooting capabilities to ensure that the system remains working and is up at all times. This study helped to describe the challenges that can be faced by a system or system administrator while monitoring a VM or data center. The tool not only carries out monitoring tasks but also provides alerts when any performance issues are encountered. This is an effective tool for monitoring as well as for failure recovery and system backup [11].

Monitoring of critical performance metrics and providing VM statistics to the user is an essential performance monitoring tool called "VM Manager Plus", which can increase the efficiency of the system by providing backup and alert notifications and help in easy recovery after failure. CPU, memory, and disk utilization, network usage, memory swap, datastore read/write latency, and a memory ready are some of the various metrics monitored by the tool [12]. If the monitored results show any discrepancy between the required and current values of the metrics, then remedial suggestions can be made and recovery of VMs can be carried out.

The Trusted Virtual Private Datacenter model was proposed for securing and managing cloud resources and services [4]. The objective was to propose a security mechanism considering the trusted relationship between the client and the IaaS provider. The model uses this relationship so as to allow both parties to set security controls to protect data and infrastructure within the cloud and virtualized data center [4].

The VES model takes into consideration the security aspects of an entire environment and suggests measures for each dimension [5]. The entire environment including the virtual machines and the virtual data centers should be secure enough to store and process data [5].

VII. CONCLUSION

This paper proposes a data management system called DMaaS to control and manage heterogeneous cloud environments. It consolidates different data centers into one single service platform to efficiently utilize system resources in the data centers. The system consists of different service layers and the main subsystems include a presentation system, business process service system, and data analysis system. Our proposed system consists of core components of resource discovery and management, role-based access control, user pattern recognition, and a recommendation system. The proposed system enables us to utilize existing data centers efficiently and effectively without extra management tools.

The proposed system is a promising service software, solving many issues related to data center consolidation. The

multi-tenant nature of this software will reduce per user cost since the fixed cost for setting up the service is shared across multiple users. Both small and large businesses can reduce their operational costs and focus on the main goal of their business rather than spending time and human resources to manage their complicated data centers.

VIII. REFERENCES

- [1] Yu-Sung Wu, Pei-Keng Sun, Chun-Chi Huang, Sung-Jer Lu, Syu-Fang Lai and Yi-Yung Chen, "EagleEye: Towards Mandatory Security Monitoring in Virtualized Datacenter Environment," 43rd Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN), June 2013.
- [2] Mohamed Esam Elsaied and Christoph Meinel, "Live Migration Impact on Virtual Datacenter Performance," IEEE International Conference on Future Internet of Things and Cloud, Barcelona, 2014.
- [3] N.Staalinprasannah and S.Suriya II, "Implementation of XenServer to ensuring business continuity through the power of virtualization for cloud computing," IEEE International Conference on Computing, Communications and Networking Technologies (ICCCNT), Tiruchengode, 2013.
- [4] Xin Wan, ZhiTing Xiao and Yi Ren, "Trusted Virtual Private Datacenter: A Model Toward Secure IaaS Cloud," Fourth International Conference on Multimedia Information Networking and Security, Nanjing, 2012.
- [5] Annette Tolnai and Sebastiaan von Solms, "A Virtualized Environment Security (VES) Model for a Secure Virtualized Environment," International Conference on Internet Technology and Secured Transactions (ICITST), London, 2010.
- [6] Fu Wen and Li xiang, "The Study on Data Security in Cloud Computing based on Virtualization", International Symposium on IT in Medicine and Education (ITME), Guangzhou, 2011.
- [7] Andre van Cleeff, Wolter Pieters, and Roel Wieringa, "Security Implications of Virtualization: A Literature Study," IEEE International Conference on Computational Science and Engineering, 2009.
- [8] Risto Vaarandi, Mauno Pihelgas, "Using Security Logs for Collecting and Reporting Technical Security Metrics," IEEE Military Communications Conference, 2014.
- [9] vSphere from VM Ware.
<http://www.vmware.com/products/vsphere>
- [10] XenServer from Citrix <http://xenserver.org/open-source-virtualization-download.html>
- [11] Veeam ONE Availability Suite.
<http://www.veeam.com/virtualization-management-one-solution.html>
- [12] VM Manager Plus.
<https://www.manageengine.com/virtualization-management/>
- [13] Netflix System Architectures for Personalization and Recommendation.
http://techblog.netflix.com/2013_03_01_archive.html
- [14] Ziyu Wang, Jiahai Yang, and Fuliang Li. "An on-line anomaly detection method based on a new stationary metric - entropy-ratio," IEEE 13th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom), Sept 2014.

A Model-Driven Approach to Generate Relevant and Realistic Datasets

Adel Ferdjoukh, Eric Bourreau, Annie Chateau, Clémentine Nebut

Lirmm, CNRS and University of Montpellier

{firstname.lastname}@lirmm.fr

Keywords: Relevant datasets generation, Meta-model instantiation, Probabilistic simulation

Abstract: Disposing of relevant and realistic datasets is a difficult challenge in many areas, for benchmarking or testing purpose. Datasets may contain complexly structured data such as graphs or models, and obtaining such kind of data is sometimes expensive and available benchmarks are not as relevant as they should be. In this paper we propose a model-driven approach based on a probabilistic simulation using domain specific metrics for automated generation of relevant and realistic datasets.

1 Introduction & Motivations

Developing software handling complex structured data requires to dispose of datasets to validate the built software. For example, when developing algorithms to assist the reconstruction of genomic sequences, sets of DNA sequences are required to experiment the algorithms. Similarly, to validate a program handling models (called a model transformation), a poll of relevant models is required. Disposing of such datasets of structured data is usually complex. Data is itself complex, and the dataset must fulfil properties such as relevancy and realism: it must contain data that look like real data. Some approaches propose an *ad hoc* generation of test cases, for instance in [10], the author presents an approach for generation of uniform lambda terms for testing a compiler of Haskell language. However, characterizing data is not fully intuitive in the proposed way.

In this paper, we propose an alternative approach for automated generation of relevant datasets. Our approach uses domain specific metrics and simulates usual probability distributions in order to generate relevant and realistic datasets. The approach is model-based: the wished data is modelled in a metamodel, and the data generation is achieved through the metamodel instantiation. Though metamodels are a strong tool to represent the abstraction underlying data, existing metamodel instantiation approaches encounter difficulties to adapt to various concrete situations, and suffer from a lack of realism in produced models. Our contribution is to inject relevance and realism into the model generation process approaching the characteristics of the desired models by the mean of *a priori* probability distributions concerning metrics about el-

ements in models.

The rest of the paper is organised as follows. Related work is presented in Section 2. Section 3 relates the simulation of probability distribution and its integration to our original approach *GRIMM*. Section 4.1 presents our first case study, the generation of real-close skeletons of Java projects. The second case study, generation of relevant scaffold graphs, is described in Section 4.2.

2 Related work

We give here an overview of contributions from the literature that are close to our proposal, i.e. in the fields of model generation through automated metamodel instantiation. Many underlying techniques have been used. *Cabot et al.* [2] translate a metamodel and its OCL constraints into constraint programming. In [9], *Mougenot et al.* use random tree generation to generate the tree structure of a model. *Wu et al.* [12] translate a meta-model into SMT (Satisfiability Modulo Theory) in order to automatically generate conform models. In [3], *Ehrig et al.* transform a meta-model into a graph grammar which is used to produce instances. The advantages and drawbacks of our original approach relatively to the other generating methods have been discussed in [4].

Nevertheless, only two approaches have treated the problem of relevance and realism of generated models. In [9], authors use a uniform distribution during the generation process and add weights in order to influence the frequency of appearance of different elements. In [12], authors describe two techniques to obtain relevant instances. The first one is the use

of partition-based criteria which must be provided by the users. The second one is the encoding of common graph-based properties. For example, they want to generate acyclic graphs, *i.e.* models.

Our goal being the generation of relevant and realistic data, we infer characteristics from real models, in a way that allows us to generate models which are close to reality.

3 Generation of relevant models

GRIMM (*GeneRATING Instances of Meta-Models*) method [4, 5] relies on translating meta-models into constraint satisfaction problems (CSP) in order to produce conform instances (models). In a nutshell, we encode the classes, references and OCL constraints of a meta-model into a set of variables connected by constraint relationships. A CSP solver performs a smart exhaustive search for values which satisfy the given constraints. Generation is fast and provides models which are guaranteed to be conform to the meta-model.

The main remaining issue is related to the usefulness of generated models. Indeed, we have to produce models that are as realistic as possible, regarding to the data it is supposed to simulate. We propose here a method that intends to achieve this goal.

The declarative approach is intrinsically deterministic, since the solver follows a deterministic algorithm to produce a unique solution. The CSP solver can easily produce thousands of solutions, but they are often far from the reality. Here we take into account the flexibility given by the CSP to encode various parameters before the solving process, and the fact that some elements of the real models follow usual probability distributions. These distributions are simulated and, *a priori*, injected to the CSP, in order to produce generated models closer to real ones.

3.1 Sampling probability distributions

Generating samples of well-known probability distributions is a way to add randomness to the deterministic CSP solving process. The idea is to get models that have more diversity in their elements' degrees and their attributes' values in order to cover a lot of possible values. For example, when generating UML models, we want to generate a package which has 5 classes, another one with 7 classes and so on.

Figure 1 shows the basic operation with which we can sample all usual probability distributions whatever they are continuous or discrete. Thus, to generate a sample of a random variable X we need its cumulative function $F(X)$ and a sample of uniform values

u . Result values x are obtained by an inversion of F : $u = F(x) \Rightarrow x = F^{-1}(u)$.

Previous method is then adapted to each probability distribution we want to sample.

Discrete distribution on a finite set For all discrete distribution, are given the probabilities of a finite set of values. The cumulative function is then deduced from the accumulation of probabilities and a sample can be easily generated.

Inverse cumulative function method This method is used for continuous distribution if their inverse cumulative function is easily computable. This method is used to simulate the exponential distribution ($\epsilon(\lambda)$).

Normal distribution: Box Muller transform Sometimes, inverting a cumulative function is difficult. In these cases, special algorithms are used. For example, a normal distribution ($\mathcal{N}(\mu, \sigma)$) does not have a known inverse function, so previous method is useless. However, many other methods exist to simulate a normally distributed sample. Our implementation uses Box Muller algorithm.

For a more complete overview about probability and simulation, please refer to [7]

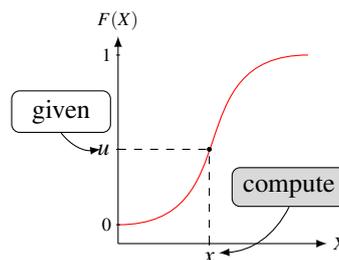


Figure 1: Simulation of random values x given a cumulative function $F(X)$ of a random variable X and uniform u

3.2 Integration into *GRIMM*

Many methods coming from the area of random graphs (see for example, [1]) use specific degree distributions to generate random graphs. Our idea is to apply such a method to randomly generate relevant models. Indeed, models are also graphs, in which vertices and edges are typed (classes and relations). There exist many domain-based metrics on the elements of models. We propose to use those metrics (viewed as probability distributions) in order to improve the relevance of randomly generated models.

The degree distribution of a link As well as in graphs, choosing the number of elements to link with a vertex (its degree) is a key to generate realistic models. The observation of real models shows us that the degrees are diverse. So the greater is the diversity of

degrees, the more realistic will be the models. In our method, the users provide probability distributions on the degree of some associations or references. Then, these distributions are simulated and integrated to the \mathcal{G} RIMM process.

Distribution on the value of an attribute The values of an attribute are also very important for the relevance of models. Indeed, generated models should have attributes with values that are close to real models. Distributions for attributes are given by the user. A probability is defined for each possible value, and simulation will choose adequate values and assign them to class instances.

Improving the connectivity Molloy and Reed show in [8] that the parametrisation of a random graph generator can influence the connectivity of the generated graphs. Indeed, choosing the adequate number of vertices and edges, and the right degree distribution gives graphs that are more connected. Our method takes into account this important aspect during the simulation process in order to get the most connected models.

Figure 2 shows the different inputs (white boxes) and the different steps (grey boxes) of our new tool \mathcal{G} YRIMM (*Generating Y*randomized and *R*elevant *I*nstances of *M*eta-Models).

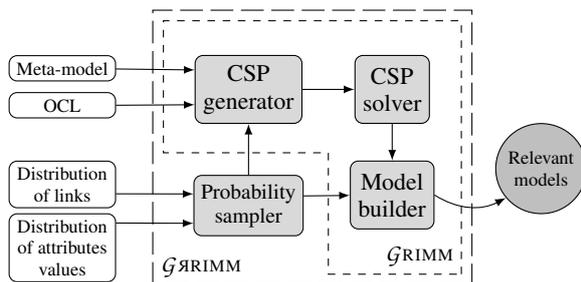


Figure 2: Methodology process.

4 Case studies

We experimentally show in this section how taking probability distributions improves the quality of generated datasets. We consider two case studies, one from Software Engineering area and the other from Bioinformatics. All data concerning the two case studies and the evaluation can be found at: <http://www.lirmm.fr/~ferdjoukh/english/experiments4Seke.html>.

4.1 Java code generation

One of the main objectives of our approach is the generation of benchmarks of test programs for different applications, such as compilers or virtual machines. In this experiment, we generate realistic and

Metric	↔	Theoretical distrib.
Class/Package	↔	$\epsilon(\frac{1}{8.387})$
Methods/Type	↔	$\epsilon(\frac{1}{7.06})$
Attributes/Type	↔	$\mathcal{N}(3.46, 2.09)$
Constructor/Type	↔	$\mathcal{N}(0.73, 0.26)$
Sub-Classes/Classe	↔	$\epsilon(\frac{1}{0.22})$
% Interface/Package	↔	$\epsilon(\frac{1}{8.001})$
Parameters/Methods	↔	$\mathcal{N}(0.87, 0.25)$

Table 1: Chosen code metrics with their theoretical probability distribution. ϵ : Exponential distribution, \mathcal{N} : Normal distribution.

relevant skeletons of Java programs using real code measurements. We choose Java for facility to find real programs to collect desired measurements. However, our method can be applied to any programming language. We collected 200 real Java projects coming from two corpus (Github and Qualitas corpus¹). For more heterogeneity, we chose projects having different sizes (big project for qualitas corpus and smaller ones from github) and different origins (well-known software such as Eclipse, Apache or ArgoUML and also small software written by only one developer). We measured metrics related to their structure, such as the percentage of concrete classes/ abstract classes, the average number of constructors for a class, the visibility of fields and methods, etc [6]. To measure these metrics we used an open source tool called Metrics². After that, we use R software³ to compute histogram of each metric in order to deduce its theoretical probability distribution. Table 1 gives the different metrics and their theoretical probability distributions.

According to these metrics, we automatically generate Java programs having the same characteristics as the real ones. To achieve this goal, we design a meta-model representing skeletons of Java projects and we adjoin some OCL constraints. 300 Java projects are generated using three versions of our approach. Four corpus are then compared: (1) projects generated by \mathcal{G} RIMM but without OCL (2) projects generated by \mathcal{G} YRIMM (3) projects generated by \mathcal{G} YRIMM (4) real Java projects. Figure 3 gives the distributions of constructors per class for each corpus. We observe that the two first versions without probability distributions give results that are very far from the characteristics of real models. On the other hand, introducing simulated probability distributions leads to substantial improvement. We see that the distribution of the number

¹Qualitas corpus: <http://qualitascorpus.com/docs/catalogue/20130901/index.html>

²Metrics tool: <http://metrics.sourceforge.net>

³R software: <https://www.r-project.org/>

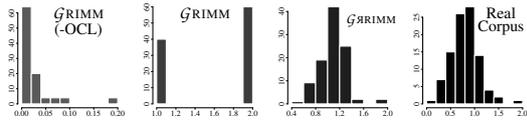


Figure 3: Comparing the number of constructors per class in Java projects. x: constructors per class, y: frequency.

of constructors of generated models are close to real ones. Moreover, these results are always better when adding probabilities for all other measurements presented in Table 1.

4.2 Scaffold graphs generation

Scaffold graphs are used in Bioinformatics to assist the reconstruction of genomic sequences. They are introduced late in the process, when some DNA sequences of various lengths, called *contigs*, have already been produced by the assembly step. Scaffolding consists in ordering and orienting the contigs, thanks to oriented relationships inferred from the initial sequencing data. A scaffold graph is built as follows: vertices represent extremities of the contigs, and there are two kind of edges. Contig edges link both extremities of a given contig (strong edges in Figure 5), whereas scaffolding edges represent the relationship between the extremities of distinct contigs. Contig edges constitute a perfect matching in the graph, and scaffolding edges are weighted by a confidence measure. Those graphs are described and used in the scaffolding process in [11] for instance. The scaffold problem can be viewed as an optimisation problem in those graphs, and consists in exhibiting a linear sub-graph from the original graph. Therefore it can be considered as well as a model transformation, when models conform to the Scaffold graph meta-model that we designed. Producing datasets to test the algorithms is a long process, somehow biased by the choices of the methods (DNA sequences generation, assembly, mapping), and there does not exist a benchmark of scaffold graphs of various sizes and densities. Moreover, real graphs are difficult and expensive to obtain. Thus, it is interesting to automatically produce scaffold graphs of arbitrary sizes, with characteristics close to the usual ones. In [11], the authors present some of these characteristics, that are used here to compare the \mathcal{G}_{RIMM} instances vs. the "hand-made" graphs.

The probability distribution chosen to produce the graph emerges from the observation that the degree distribution in those graphs is not uniform, but follows an exponential distribution. We compare several datasets, distributed in several classes according to their sizes: (1) graphs generated by \mathcal{G}_{RIMM} , (2)

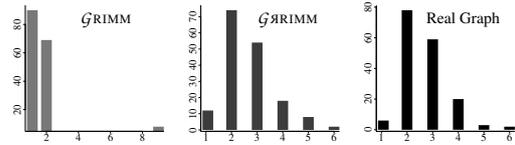


Figure 4: Comparing the degree distribution between a real scaffold graph and its equivalent generated ones (168 nodes and 223 edges).x: degree, y: frequency.

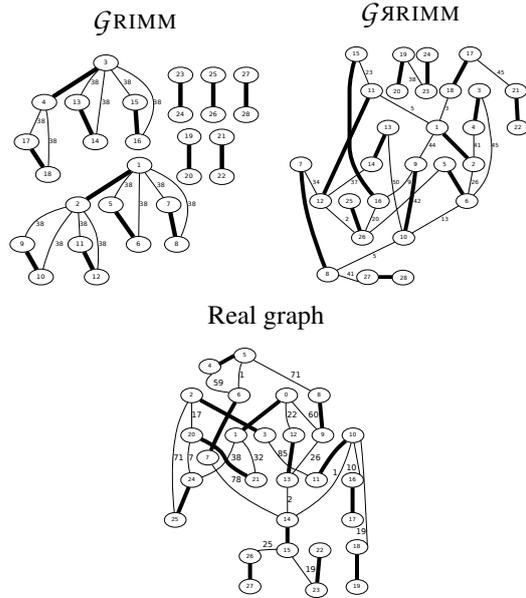


Figure 5: Three Scaffold graphs corresponding to the same species (monarch butterfly). Strong edges represent contig edges, other edges are scaffolding edges.

graphs produced by \mathcal{G}_{RIMM} and (3) real graphs of different species, described in [11].

For each real graph, 60 graphs of the same size are automatically generated. 30 graphs are naively generated using the original \mathcal{G}_{RIMM} method [4, 5], and 30 others are generated after the simulating of probability distribution. These models are then compared in term of visual appearance (Figure 5), degree distribution (Figure 4) and according to some graph measurements (Table 2).

We see in Figure 5 three models (scaffolds graphs) corresponding to the same species (monarch butterfly⁴). The naive method generates a graph that does not look like the real one. This graph is too weakly connected, and the connected parts have a recurring pattern. This is not suitable for a useful scaffold graph. Whereas, introducing probabilities provides graphs having shapes close to reality. Thus, both real graphs and generated graphs (with probability distri-

⁴It refers to mitochondrial DNA of monarch butterfly.

Graph size			Measurements					
			\mathcal{G} GRIMM generation		\mathcal{G} ЯRIMM generation		Real graphs	
Species	nodes	edges	min/max degree	h-index	min/max degree	h-index	min/max degree	h-index
monarch	28	33	1/9	3	1/ 4.6	4.06	1/ 4	4
ebola	34	43	1/9	3	1/ 4.83	4.60	1/ 5	4
rice	168	223	1/9	8	1/ 6.03	5.93	1/ 6	5
sacchr3	592	823	1/9	10	1/ 7	6.76	1/ 7	6
sacchr12	1778	2411	–	–	1/ 7.53	7	1/10	7
lactobacillus	3796	5233	–	–	1/ 8.06	7.8	1/12	8
pandora	4092	6722	–	–	1/ 8.23	7.96	1/ 7	7
anthrax	8110	11013	–	–	1/ 8.3	8.03	1/ 7	7
gloeobacter	9034	12402	–	–	1/ 8.46	8	1/12	8
pseudomonas	10496	14334	–	–	1/ 8.43	8	1/ 9	8
anopheles	84090	113497	–	–	1/ 8.96	9	1/ 51	12

Table 2: Comparing graph metrics on real scaffold graphs and average for 60 generated ones for each species.

bution) are strongly connected, and more randomness can be observed in the connections and the weights of edges.

Figure 4 compares the degree distributions for three scaffold graphs of the same species. We see that generating with probabilities gives a distribution very similar to the distribution in the real graph.

Table 2 compares the three benchmarks of scaffold graphs (naive generation, generation with probabilities and real graphs) according to some graph measurements. We can observe again, that the graphs generated with probabilities are closer to real graphs than the naively generated graphs in all cases. The measurements on the naive graph suffer from a lack of diversity and randomness. Indeed, the minimal and the maximal degree are the same for all generated models. This, of course, does not reflect the reality. Notice also that it was not possible with the naive generation method to generate largest graphs corresponding to largest genomes.

5 Conclusion & Discussion

In this work, we presented a model driven approach to generate realistic and useful datasets. Our approach exploits domain-based metrics and the simulation of probability distributions to tackle the issue of relevance for generated instances. We evaluated our work by applying the method to two different fields: generation of source code skeletons in Software Engineering and generation of scaffold graphs in Bioinformatics. The method follows four main steps: (1) Design a meta-model to represent the domain, (2) Collect metrics on real models (The metrics can also be related to a specific use, so given by an expert), (3) Sample probability distributions with respecting previous metrics and (4) Generate realistic and relevant instances using \mathcal{G} ЯRIMM tool. We observed a substantial improvement of relevance when simulated

probabilities samples are added to the model generator. New generated instances have characteristics very close to real models, improving their usefulness for testing programs. This is especially interesting when data is rare, difficult or expensive to obtain, like in the case of scaffold graphs.

REFERENCES

- [1] B. Bollobás. *Random Graphs*. Cambridge University Press, 2nd edition, 2001.
- [2] J. Cabot, R. Clarisó, and D. Riera. Verification of UML/OCL Class Diagrams using Constraint Programming. In *IEEE ICSTW*, pages 73–80, 2008.
- [3] K. Ehrig, J. M. Kister, G. Taentzer, and J. Winkelmann. Generating Instance Models from Meta Models. In *FMOODS*, pages 156–170, 2006.
- [4] A. Ferdjouxh, A.-E. Baert, E. Bourreau, A. Chateau, R. Coletta, and C. Nebut. Instantiation of Meta-models Constrained with OCL: a CSP Approach. In *MODELSWARD*, pages 213–222, 2015.
- [5] A. Ferdjouxh, A.-E. Baert, A. Chateau, R. Coletta, and C. Nebut. A CSP Approach for Metamodel Instantiation. In *IEEE ICTAI*, pages 1044–1051, 2013.
- [6] B. Henderson-Sellers. *Object-Oriented Metrics: Measures of Complexity*. Prentice Hall, 1996.
- [7] M. Mitzenmacher and E. Upfal. *Probability and Computing: Randomized Algorithms and Probabilistic Analysis*. Cambridge University Press, 2005.
- [8] M. Molloy and B. Reed. A Critical Point for Random Graphs with a Given Degree Sequence. *Random Structures & Algorithms*, 6(2-3):161–180, 1995.
- [9] A. Mougnot, A. Darrasse, X. Blanc, and M. Soria. Uniform Random Generation of Huge Metamodel Instances. In *ECMDA*, pages 130–145, 2009.
- [10] M. Palka. *Random Structured Test Data Generation for Black-Box Testing*. PhD thesis, University of Göteborg, 2014.
- [11] M. Weller, A. Chateau, and R. Giroudeau. Exact approaches for scaffolding. *BMC Bioinformatics*, 16(14):1471–2105, 2015.
- [12] H. Wu, R. Monahan, and J. F. Power. Exploiting Attributed Type Graphs to Generate Metamodel Instances Using an SMT Solver. In *TASE*, pages 175–182, 2013.

A Query Language of Data Provenance Based on Dependency View for Process Analysis

Xuan Sun

Beijing Information Science
& Technology University
Peking, China
sunxuanbupt@126.com

Xin Gao

The National Computer
network Emergency
Response technical Team
Coordination Center of
China
Peking, China
gaoxin54@126.com

Huiying Du

Beijing Information Science
& Technology University
Peking, China
huiyingdu@bistu.edu.cn

Wei Ye

National Engineering
Research Center for
Software Engineering,
Peking University
Peking, China
wye@pku.edu.cn

Abstract—For the scale of data in process keep increasing, data provenance also becomes large and constantly growing, which brings challenges to the efficiency of provenance tracking in process analysis. This paper proposes a kind of dependency view to extract a global data provenance description of the data process instance, and then defines a contextual query language based on dependency view to implement an efficient provenance query mechanism for process analysis. The elements of the language are based on a set of dependency view query operations, which can decrease the steps of provenance tracking based on the elements of data provenance and support the descriptive power of the language for complex provenance tracking. Experimental results show that complex provenance tracking by the language is efficient and ease to use.

Keywords—provenance; query language; dependency view

I. INTRODUCTION

Data provenance records the related operations and data in the execution of data process, which is regarded as an important data in many process-aware systems. In the application domains of data provenance, it is used for auditing in the data base and supporting analysis in the complex science experiment environment in the early times, process analysis and process verification following the development of workflow, semantic resolving in the web, and now existing as metadata in the cloud environment [1, 2, 3]. Following the development of the application of data provenance, we can find out that the scale of the data object supported by data provenance becomes more and more huge. Therefore it directly increases the amount of the intermediate data in the process of the source data. So now we need to satisfy the efficient requirement of data provenance analysis. Currently, many researchers try to deal with this problem based on distributed data management platform, like cloud platform, which takes the advantage of large-scale storage and computing power of cloud platform. Ikeda et al. [4] propose an approach for tracking the provenance of workflow modeled as MapReduce jobs. Malik et al. [5] introduce an approach for recording provenance in distributed environment and each node stores parts of entire provenance graph. Based on cloud environment, it can temporarily increase the efficiency of data provenance

query, but its cost becomes higher when the scale of data provenance keep increasing and meanwhile it still needs to program the query function in the cloud environment. So we finally still need to optimize the query mechanism of data provenance when the query requirement becomes more and more complex as the data provenance dataset keeps growing. However, current data provenance models are always not ready for directly implement the query requirements, because they aim at describing the dependency relations among the elements of data provenance. So extracting the dependency relations from data provenance and providing higher view of data provenance is a way to improve the efficiency of data provenance query. In this paper, we focus on the efficient and usable querying of data provenance and carry out our research from the aspect of data provenance description and corresponding query mechanism. We try to extract dependency elements from data provenance directed graphs to form the dependency view and define corresponding query operations of dependency view for provenance tracking, which can make data provenance records suit to be queried. Then we propose a query language based on dependency view to describe the requirement of complex provenance tracking.

The remainder of this paper is organized as follows. We introduce the related work in section 2, introduce dependency view of data provenance in section 3, define the query operations for data provenance based on dependency view in section 4, propose the data provenance query language in section 5, and validate the efficiency of query language in section 6. In the end, we make our conclusions in section 7.

II. RELATED WORK

Data provenance as the key technology for data-intensive research provides a kind of causal relationship model among the result, operation, middle data, and human and so on. There has been significant progress on formal models for data provenance. However, the difficulty to support process analysis based on data provenance is mainly the complexity and variety of the analysis requirement, which proposes a serious description problem for data provenance querying. Then the issue of data provenance querying is just addressed in a application-independent way to in recent years, the query

languages of data provenance are proposed to resolve the problem, like OPQL [6], VQuel [7], ProQL [8], QLP [9], etc. These query languages are based on formal models for data provenance, which cause the difficulty in writing query by these languages. Therefore, we need a flexible and extensible query language of data provenance to support dependency deducing of data provenance while the complex and various requirements of process analysis are proposed.

III. DEPENDENCY VIEW OF DATA PROVENANCE

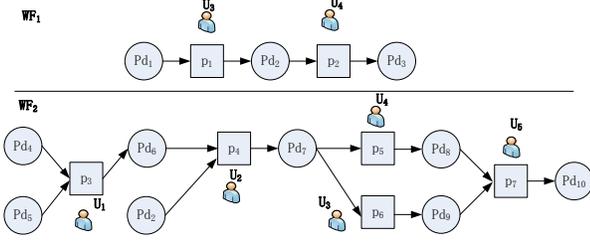


Figure 1. An execution instance of data process

For each task in the process execution, its dependency elements always include the related input data, tasks, operators and constraint, just as figure 1. All these dependencies consist of the context for the executing task, which can be extracted as directed graph from the executed part of the instance. We call these extracted directed graphs as dependency view of execution instance. According the type of the node in dependency view, we divide them into three categories: data dependency view, process dependency view and collaboration dependency view. The data dependency view is signed as **DFDepView**= $\langle \mathbf{D}, \mathbf{DFDep} \rangle$. **D** stands for data set of input data and output data, and **DFDep** stands for the casual relationships of “was Derived From”. The process dependency view is signed as **PFDepView**= $\langle \mathbf{P}, \mathbf{PFDep} \rangle$. **P** stands for the task instance set, and **PFDep** stands for “was Triggered By”. The collaboration dependency view is signed as **HFDepView**= $\langle \mathbf{H}, \mathbf{HFDep} \rangle$. **H** stands for operator set, and **HFDep** stands for the collaboration relationships among **H**. So we use an abstract model $\langle \mathbf{N}, \mathbf{E} \rangle$ to describe these three dependency views, where **N** stands for the nodes in dependency view and **E** stands for the dependency relationship in dependency view. Assuming that **A** and **B** belong to the same kind of dependency view, and the operations for dependency view are defined as table I.

TABLE I. DESCRIPTION OF OPERATIONS FOR DEPENDENCY VIEW

Operation	Description
$\mathbf{A} \cup \mathbf{B} = \langle \mathbf{A.N} \cup \mathbf{B.N}, \mathbf{A.E} \cup \mathbf{B.E} \rangle$	union operation for dependency view
$\mathbf{A} \cap \mathbf{B} = \langle \mathbf{A.N} \cap \mathbf{B.N}, \mathbf{A.E} \cap \mathbf{B.E} \rangle$	intersection operation for dependency view
$\mathbf{A} - \mathbf{B} = \langle \mathbf{A.N} - \mathbf{B.N}, \{e \in (\mathbf{A.E} - \mathbf{B.E}) \wedge e.source \in (\mathbf{A.N} - \mathbf{B.N}) \wedge e.destination \in (\mathbf{A.N} - \mathbf{B.N})\} \rangle$	complement operation for dependency view
$\mathbf{A} \oplus \mathbf{B} = \langle \mathbf{A.N} \oplus \mathbf{B.N}, \{e \in (\mathbf{A.E} \oplus \mathbf{B.E}) \wedge e.source \in (\mathbf{A.N} \oplus \mathbf{B.N}) \wedge e.destination \in (\mathbf{A.N} \oplus \mathbf{B.N})\} \rangle$	symmetric difference operation for dependency view

Beside these basic operations, there are also the operations for the set of dependency view. Assuming the set **A** consists of $\mathbf{A}_1, \mathbf{A}_2, \dots, \mathbf{A}_n$ which belong to the same kind of dependency view, signed as $\mathbf{A} = \{\mathbf{A}_1, \mathbf{A}_2, \dots, \mathbf{A}_n\}$. Then the corresponding union and intersection operation can be carried out as following:

$$\cup \mathbf{A} = \mathbf{A}_1 \cup \mathbf{A}_2 \cup \dots \cup \mathbf{A}_n; \cap \mathbf{A} = \mathbf{A}_1 \cap \mathbf{A}_2 \cap \dots \cap \mathbf{A}_n$$

From the point view of dependency view, any OPM instance can transform into the three dependency views and can be analyzed based on the operations of dependency view. Thus dependency view provides a more coarse-grained description than OPM[10], which provides a base for the improvement of data provenance tracking. So we describe the context of the task at runtime based on dependency view in this paper, and try to deduce the requirement of provenance tracking based on dependency view. Assuming task *p* is going to be executed in the data process execution instance and the input data of *p* is the data set $\{d_1, \dots, d_n\}$, we use $\text{PGrap}(d)$ to describe the data provenance of data *d*, and the data provenance of the process execution instance before *p* executes can be described as $\text{InputP} = \text{PGrap}(d_1) \cup \dots \cup \text{PGrap}(d_n)$.

Definition (The context of the task at runtime): the context for task *p* is described as following:

$$\text{Context}(p) = \{ \text{Constraints}(p), \mathbf{DFDepView}(\text{InputP}), \mathbf{PFDepView}(\text{InputP}), \mathbf{HFDepView}(\text{InputP}) \}$$

IV. THE QUERY OPERATIONS FOR DATA PROVENANCE BASED ON DEPENDENCY VIEW

A. Basic query operation

TABLE II. DESCRIPTION OF BASIC QUERY OPERATION

Operation	Query form	Description
Q ₁	$\{p\} \dots Pd$	What tasks directly or indirectly involve in the production of data <i>Pd</i> ?
Q ₂	$\{Pd\} \dots p$	What data directly or indirectly affect the execution of task <i>p</i> ?
Q ₃	$\{h\} \dots p$	Who joins the collaboration triggering task <i>p</i> ?
Q ₄	$\{p'\} \dots p$	What tasks directly or indirectly trigger task <i>p</i> ?
Q ₅	$\{Pd'\} \dots Pd$	What data directly or indirectly involve in the production of data <i>Pd</i> ?
Q ₆	$\{h'\} \dots h$	Who directly or indirectly joins the collaboration with operator <i>h</i> ?
Q ₇	$\{p\} \dots h$	What tasks directly or indirectly affect operator <i>h</i> ?
Q ₈	$\{h\} \dots Pd$	Whose collaborations directly or indirectly affect the production of data <i>Pd</i> ?
Q ₉	$\{Pd\} \dots h$	What data are directly or indirectly used by operator <i>h</i> ?

Based on the dependency views of the task at runtime, the formal expressions of these basic query operations in table II can be expressed as table III.

TABLE III. FORMAL EXPRESSIONS OF BASIC QUERY OPERATION

Operation	Formal expression
Q ₁ (<i>Pd</i>)	$\cup \{ \text{Context}(p), \mathbf{PFDepView} \cup \{p\} \}$, where $p \in \{ p' \mid (\text{Generatedby}: Pd \rightarrow p') \}$
Q ₂ (<i>p</i>)	$\text{Context}(p), \mathbf{DFDepView}$
Q ₃ (<i>p</i>)	$\text{Context}(p), \mathbf{HFDepView}$
Q ₄ (<i>p</i>)	$\text{Context}(p), \mathbf{PFDepView}$

$Q_5(Pd)$	$\cup \{ \text{Context}(p). \mathbf{DFDepView} \},$ where $p \in \{ p' \mid (\text{Used}: p' \rightarrow Pd) \}$
$Q_6(h)$	$\cup \{ \text{Context}(p). \mathbf{HFDepView} - h, \}$ where $p \in \{ p' \mid (\text{Controlledby}: p' \rightarrow h) \}$
$Q_7(h)$	$\cup \{ \text{Context}(p). \mathbf{PFDepView} \},$ where $p \in \{ p' \mid (\text{Controlledby}: p' \rightarrow h) \};$
$Q_8(Pd)$	$\cup \{ \text{Context}(p). \mathbf{HFDepView} \cup \{ h \} \},$ where $p \in \{ p' \mid (\text{Generatedby}: Pd \rightarrow p') \} \wedge (\text{Controlledby}: p \rightarrow h);$
$Q_9(h)$	$\cup \{ \text{Context}(p). \mathbf{DFDepView} \},$ where $p \in \{ p' \mid (\text{Controlledby}: p' \rightarrow h) \}$

B. Influence query operation

TABLE IV. DESCRIPTION OF INFLUENCE QUERY OPERATION

Operation	Query form	Description
DataAffectQ ₁	$Pd \dots \{ Pd' \}$	What data directly or indirectly are affected the production of data Pd' ?
DataAffectQ ₂	$Pd \dots \{ p \}$	What tasks directly or indirectly are affected by data Pd ?
DataAffectQ ₃	$Pd \dots \{ h \}$	Whose collaborations are directly or indirectly are affected by data Pd ?
ProcessAffectQ ₁	$p \dots \{ p' \}$	What tasks directly or indirectly are affected by task p' ?
ProcessAffectQ ₂	$p \dots \{ Pd \}$	What data directly or indirectly are affected by task p ?
ProcessAffectQ ₃	$p \dots \{ h \}$	Whose collaborations are affected by task p ?
HumanAffectQ ₁	$h \dots \{ h' \}$	Whose collaborations are affected by h ?
HumanAffectQ ₂	$h \dots \{ Pd \}$	What data are directly or indirectly affected by the collaborations with h ?
HumanAffectQ ₃	$h \dots \{ p \}$	What tasks are directly or indirectly affected by the collaborations with h ?

Assuming **PDepViewofAll**, **DDepViewofAll** and **CDepViewofAll** stands for the process dependency view, data dependency view and collaboration dependency view of the whole process execution instance, the formal expressions of basic query operations in table IV can be expressed as table V.

TABLE V. FORMAL EXPRESSIONS OF INFLUENCE QUERY OPERATION

Operation	Formal expression
DataAffectQ ₁ (Pd)	DDepViewofAll - Q ₅ (Pd)
DataAffectQ ₂ (Pd)	PDepViewofAll - Q ₁ (Pd)
DataAffectQ ₃ (Pd)	CDepViewofAll - Q ₈ (Pd)
ProcessAffectQ ₁ (p)	PDepViewofAll - Q ₄ (p)
ProcessAffectQ ₂ (p)	DDepViewofAll - Q ₂ (p)
ProcessAffectQ ₃ (p)	CDepViewofAll - Q ₃ (p)
HumanAffectQ ₁ (h)	CDepViewofAll - Q ₆ (h)
HumanAffectQ ₂ (h)	DDepViewofAll - Q ₉ (h)
HumanAffectQ ₃ (h)	PDepViewofAll - Q ₇ (h)

V. QUERY LANGUAGE

Though query operations based on dependency view has improve the efficient of data provenance tracking, we still need to deal with the complexity of the transformation from the requirements of data provenance analysis to the query operations. Therefore we propose a contextual query language base on these query operations.

A. Data set based on dependency view

Dependency view is the basic element for the context of task at runtime, the result of basic query operation and the result of influence query operation. Therefore the data related to data provenance tracking is based on dependency view

which can be seen as a kind of virtual data table in this paper, and any operation based on dependency view is actually the query to these virtual data tables. Therefore basic query operations, influence query operations and specific dependency views can be divided into three categories from the point view of dependency view, which can carry out the operations to one another internally. The partitions are show as table VI, in which "Instance" stands for the process execution instance.

TABLE VI. DATA SET BASED ON DEPENDENCY VIEW

Dependency view	Corresponding data set
Data dependency view	Instance.Context().DDepViewofAll; Instance.Context(p).DFDepView; Q ₂ , Q ₅ and Q ₉ ; DataAffectQ ₁ , ProcessAffectQ ₂ and HumanAffectQ ₂
Process dependency view	Instance.Context().PDepViewofAll; Instance.Context(p).PFDepView; Q ₁ , Q ₄ and Q ₇ ; ProcessAffectQ ₁ , DataAffectQ ₂ and HumanAffectQ ₃
Collaboration dependency view	Instance.Context().CDepViewofAll; Instance.Context(p).HFDepView; Q ₃ , Q ₆ and Q ₈ ; HumanAffectQ ₁ , DataAffectQ ₃ and ProcessAffectQ ₃

B. Operators for data set based on dependency view

Product operation merges one dependency view to another dependency view by eliminating duplicate nodes and edges. Assuming the product of data set t_1 and t_2 is signed as $t_1 \times t_2$, product operation between t_1 and t_2 is described as follow:

$$t_1 \times t_2 = \langle \mathbf{N} \cup \mathbf{N}', \mathbf{E} \cup \mathbf{E}' \rangle$$

where the corresponding direct graph of t_1 is $\langle \mathbf{N}, \mathbf{E} \rangle$ and the corresponding direct graph of t_2 is $\langle \mathbf{N}', \mathbf{E}' \rangle$.

Selection operation selects out the tuples satisfied predicate, which use σ to stand for selection operation and set predicate to the subscript of σ . For the data set based on dependency view, there are two differences from relation database. First, the variables in predicate must be the variables defined in dependency view. These variables include: node (standing for data, task and human), edge (standing for the dependency like "Generatedby", "Used", "Triggerredby" and so on), and subgraph (standing for the part of dependency view satisfying specific constraint). Second, it only permit to use the operators including $=$, \in and \subseteq . The operators likes \neq , \leq , \geq , $<$ and $>$ are refused. But it can compose the bigger predicate by the operators of single predicate, like \wedge , \vee and \neg . For example, to find the process nodes affected by both p_7 and Pd_2 in figure1 can be described as follow:

$$\sigma_{\text{subGraph} \subseteq WF_2.P_7.Context.PFDepView \wedge \text{subGraph} \subseteq \text{DataAffectQ}_2(WF_2.Pd_2)} (WF_2.Context.PFDepViewAll)$$

Projection operation for data set based on dependency view can be signed by Π . The subscript of Π consists of node, edge and sub graph, meanwhile the rear parameters consists of the data set and the corresponding operators. For example, to find out the tasks which depend on task p_7 can be described as follow:

$$\Pi_{\text{node}} WF_2.p_7.Context.PFDepView$$

C. Syntax of contextual query language

As SQL of relation database, the contextual query language also includes three clauses: select, from and where. Corresponding to projection operator, select clause is used to get the specific elements from process execution instance to satisfy the requirement of data provenance analysis. These elements which are part of context of the task at runtime include: the node (standing for data, task and human), the edge (standing for the dependency) and the sub graph. Corresponding to product operator, from clause is used to list the data set based on dependency view related to the requirement of data provenance analysis. Corresponding to the predicate of selection operator, where clause is used to describe the condition and the constraint for filtering the data set based on dependency view satisfied the requirement of data provenance analysis. Referring to the structure of relation database query language, the many-to-many query of data provenance based on context can be abstracted as follow:

Select c_1, c_2, \dots, c_n **From** $\text{DepView}_1, \text{DepView}_2, \dots, \text{DepView}_m$
Where predicates

DepView_j stands for the data set based on dependency view, and c_i stands for the node, edge, or sub graph in the data set. Therefore the query statement of contextual query language is equivalent to the corresponding data set based on dependency view with their operators, and many-to-many query can also be described as follow:

$$\prod_{c_1, c_2, \dots, c_n} (\sigma_{\text{predicates}} (\text{DepView}_1 \times \text{DepView}_2 \times \dots \times \text{DepView}_m))$$

VI. EXPERIMENT

To validate the advantage of our query mechanism based on dependency view to the query based on OPM at the aspect of the data provenance tracking, we assess the performance of data provenance query as follow: Firstly, select one node from WF_2 randomly, and then execute query related to the node, which is seen as one test case. Secondly, collect the time cost through the execution of test case as the performance of the query statement. We use the number of database access to stands for the time cost, because the time cost of provenance tracking mainly spend on database access and each database access cost the similar time. Thirdly, set the parameter n as the least number of test case executions, carry out the executions of test case at n times, and increase by $10n$ times. Finally, sort out the data of performance index of test execution by the parameter n , and then sum the data with the same parameter n .

In this test, we select the complex query requirement “find out the tasks that are affected by the data Pd_2 and also trigger the task p_7 ” as the test object. Meanwhile, our query language can describe the query requirement as follow query statement:

Select node **From** $p_7.\text{Context.PFDepView} \cap \text{DataAffectQ2}(Pd_2)$

Where the cost of this statement consists of the cost of $p_7.\text{Context.PFDepView}$ and the cost of $\text{DataAffectQ2}(Pd_2)$.

Set 5 to the parameter n , and the result is shown in figure 2, where the ordinate is the number of database access and the abscissa is the number of the increment of n . In figure 2, the red line stands for the result of query statement, and the blue line stands for the result of regular query based on OPM. We

can see that the result of query statement is bigger than the result of the regular query at the beginning, but soon the result of query statement is smaller than the result of the regular query and the gap continues to become wider. The reason is that the dependency view needs more database access than regular query based on OPM to extract the dependency view from the whole data provenance directed graph at the beginning, and then it can support any basic query operation by just one database access. Therefore the experiment shows that the query language based on dependency view is more efficient.

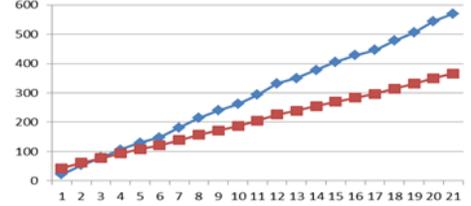


Figure 2. The performance comparing between query statement and corresponding regular query based on OPM

VII. CONCLUSION

In this paper, we study the query language for the data provenance which keeps growing as the development of data intensive systems and process-aware system. Through the study, we review the challenge and requirement to current data provenance query, and try to find a new language to solve the process analysis problem base on dependency view of data provenance. In the future, we will continue our work to support more and more complex requirements of process analysis based on data provenance.

ACKNOWLEDGEMENTS

Xin Gao is the corresponding author of this paper.

REFERENCES

- [1] R. Bose and J. Frew. Lineage retrieval for scientific data processing: a survey, *ACM Comput. Surv.*, 37(1): 1-28, 2005.
- [2] Y. Simmhan, B. Plale, and D. Gannon. A survey of data provenance in e-science, *SIGMOD Record*, 34(3): 31-36, 2005.
- [3] Muniswamy-Reddy, K., Seltzer, M., Provenance as First-Class Cloud Data, 3rd ACM SIGOPS International Workshop on Large Scale Distributed Systems and Middleware (LADIS'09), October 2009.
- [4] Ikeda, R., Park, H., Widom, J.: Provenance for generalized map and reduce workflows. In: *CIDR*. pp. 273-283 (2011)
- [5] Malik, T., Nistor, L., Gehani, A.: Tracking and sketching distributed data provenance. In: *eScience*. pp. 190-197, 2010.
- [6] Chunhyeok Lima, Shiyong Lua, Artem Chebotkov, Farshad Fotouhnia, Andrey Kashleva, *OPQL: Querying scientific workflow provenance at the graph level*, *Data & Knowledge Engineering*, Volume 88, Pages 37-59, November 2013
- [7] Amit Chavan, Silu Huang, Amol Deshpande, Aaron Elmore, Samuel Madden, Aditya Parameswaran, *Towards a Unified Query Language for Provenance and Versioning*, *International Workshop on Theory and Practice of Provenance*, 2015.
- [8] Grigoris Karvounarakis, Zachary G. Ives, Val Tannen, *Querying Data Provenance*, *SIGMOD*, pages: 951-962, 2010
- [9] M. K. Anand, S. Bowers, and B. Ludächer. Techniques for efficiently querying scientific workflow provenance graphs. In *EDBT*, volume 10, pages 287-298, 2010.
- [10] Moreau L, Freire J, Futrelle J, et al. *The open provenance model*, Southampton: School of Electronics and Computer Science, University of Southampton, 2007.

Collecting Usage Data for Software Development: Selection Framework for Technological Approaches

Sampo Suonsyrjä, Kari Systä, Tommi Mikkonen and Henri Terho
Tampere University of Technology, Korkeakoulunkatu 1, FI-33720 Tampere, Finland
{sampo.suonsyrja, kari.systa, tommi.mikkonen, henri.terho}@tut.fi

Abstract—Software development methods are shifting towards faster deployments and closer to the end users. Their ever tighter engagement of end-users also requires new technologies for gathering feedback from those users. At the same time, widespread Internet connectivity of different application environments is enabling the collection of this post-deployment data also from sources other than traditional web and mobile software. However, the sheer number of different alternatives of collecting technologies makes the selection a complicated process in itself. In this paper, we describe the process of data-driven software development and study the challenges organizations face when they want to start guiding their development towards it. From these challenges, we extract evaluation criteria for technological approaches to usage data collecting. We list such approaches and evaluate them using the extracted criteria. Using a design science approach, we refine the evaluation criteria to a selection framework that can help practitioners in finding a suitable technological approach for automated collecting of usage data.

I. INTRODUCTION

One of the clear trends in the field of software development has been the ever tighter engagement of the end-users to the software development process. For example, methods such as Lean Startup [1] are dependent on more and more rapid feedback cycles. As described in a more general level in [2], the shift from Agile processes towards Continuous Deployment and experiment systems requires faster ways to validate the developed software than is possible with traditional communication methods, such as face to face conversations with end-users.

As these new methods are emerging, the whole software development process can be rearranged. In the aforementioned experiment systems for example, the deployment of software is not the end of the road for development efforts, but more of an initial step to start collecting data on user needs and then fine-tune the software [2]. With such approach, post-deployment data is first collected and then used for guiding the software development making the development process data-driven.

First and foremost this post-deployment data, such as data about how the system is used (i.e. usage data), has been used for guiding software development in environments like web and mobile development. In these contexts, constant connectivity – an important enabler for usage data collection – is the norm. However, breakthroughs of cloud software and Software-as-a-Service model, and the fact that most applications and platforms are Internet connected to begin with,

are extending the use of data collection to a wider range of applications.

As this range of potential target applications, or programs whose usage data can be collected from, is getting wider, we are left with the challenge of finding the right technological approach for usage data collecting in the varying target application environments and cases. For example, manually adding code to target applications for logging purposes can be a straightforward option for developers in simple cases on one hand. But on the other hand, there are also different kinds of standardized tools and various approaches that among other things can automate this instrumentation or at least some parts of it.

To address this, we study what kind of challenges organizations face when they are starting the usage data collecting. Literature reviews along with a case study in an international telecommunication organization are used for finding these challenges, and they are extracted into evaluation criteria for data collecting technologies. We then describe several options for the automatic usage data collecting and evaluate them with the formed criteria. After this, we refine the criteria and the evaluated technological approaches into a selection framework that should help practitioners choose the suitable technologies.

The main research problem is *how to select the right technological approach for automated collecting of usage data?*. To address this, we derive two research questions from the main problem as follows.

- RQ1: How to evaluate different technological approaches for automated collecting of usage data?
- RQ2: What kind of technological approaches are there for the automated collecting of usage data?

The rest of the paper is structured as follows. In Section II, we take a look at the context of data-driven software development. Additionally, we go through the appropriate literature to find out challenges in automatic collecting of post-deployment data. Section III explains the formed evaluation criteria, and in Section IV we use the criteria to evaluate several technological approaches to usage data collecting. In Section V we derive a selection framework from the evaluation criteria and the technological approaches. In Section VI we draw some final conclusions.

II. BACKGROUND

The background of this paper is two-fold. First, we address data-driven software development. Then, we introduce the

challenges of automatic usage data collecting.

A. Data-Driven Software Development

As presented in [2], companies typically evolve their software development processes by climbing the *Stairway to Heaven* (StH). StH describes the shift from traditional waterfall development towards continuous deployment of software. The steps to be taken in the proposed chronological order are *Traditional Development*, *Agile R&D Organization*, *Continuous Integration*, *Continuous Deployment*, and the model ends up with *R&D as an Experiment System*. With each step, software development is becoming faster in the sense that it produces new releases of software ever more quickly. In the scope of this paper, the last phase is especially interesting as climbing the last step requires a fast-track of information from customers back to the development organization.

However, feedback gathering from customers is often slow, and sufficient mechanisms for it are missing. This can result in opinion-based development decisions. To ease the climb to the final step and make development more data-driven, Olsson & Bosch have developed the HYPEX model, i.e. *Hypothesis Experiment Data-Driven Development* [3]. In this model, *Minimal Viable Features* (MVF) are implemented and their expected behavior is described. A feature is implemented over many iterations and so the first 10% to 20% of its functionality is called an MVF. The deployed MVF is always instrumented to collect data on its use by customers, and this data is then compared with the initial descriptions of how the development organization thought that it would be used. Based on this *Gap Analysis*, the developers then either finalize or abandon the feature, or iterate the experimentation over again with a different hypothesis.

Such process has a lot in common with the *Build-Measure-Learn loop* (BML-loop) described in [1]. Compared to the HYPEX model, the BML loop has many similarities and main differences are in the abstraction level. The BML loop is meant to validate the business feasibility of the product through the use of *Minimum Viable Products* (MVP). During each turn of the BML-loop a product hypothesis is formed and measurable metrics are linked to the hypothesis. The MVP is then built and the metrics are measured. Based on the outcomes of this data, the decision is made if the product development should be continued or another product hypothesis should be tested based on the experiences learned from the MVP.

As described above, both the HYPEX model and the BML-loop are data-driven approaches to software development. *Software Analytics*, as laid out in [4], highlights this use of data as well. However, this paradigm of analytics points out specifically the different types of analyses that are needed for turning the measured data into insights and eventually into development decisions. These are depicted in Figure 1. The model originates from the field of web analytics, but as its generality seems broad and with its experimental approach it should suit organizations well as a guidance in the *R&D as an Experiment System* phase of StH.

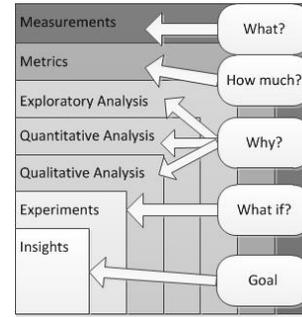


Fig. 1. Paradigm of Analytics (adapted from [4]).

All the aforementioned models include the phases of planning the data collecting, collecting the data, and analyzing the results to make decisions and iterating the process over again. In this sense, data-driven software development can be seen as an overarching term that typically consists of similar phases. To get a concrete definition from a technology standpoint and in the scope of this paper, we have formulated *Data-Driven Software Development* as an iterative process as follows.

- 1) *Planning of the data collection.* The goals of the analysis need to be known and the monitored applications and features should be selected based on them. The required resources, customer and user permissions and legal aspects of data collection need to be checked as well.
- 2) *Deployment of data collection.* The infrastructure of technical means to track the applications and collect post-deployment data needs to be installed.
- 3) *Monitoring of the applications.* The technical means can be internal to the application but also external - depending on the used run-time and platform technologies.
- 4) *Picking up the relevant data.* Monitoring should be configured to pick the data that is seen useful for the planned data collection and analysis.
- 5) *Pre-processing – filtering and formatting – the data.* The collected data is typically transferred to a remote location, but is typically filtered and formatted before sending to save resources.
- 6) *Sending and/or saving the data.* For effective analysis the data needs to be collected from long enough period and it needs to be available for the people working on the analysis. Often this means that hosting of the data storage is different from the applications. Thus, the system should transfer the data to storage either by means of continuous streaming or by saving it first to local cache and sending bigger amounts of data at the same time.
- 7) *Cleaning and unification of the data.* This process completes the work done by pre-processing described earlier but is necessary especially if data is flowing from various different sources.
- 8) *Storing the data.* Typically some database is used for storing the data.
- 9) *Visualizations and analysis.* A tools set helps stakehold-

ers to ask "what" and "how much" questions and to make conclusions.

- 10) *Decision making.* The results should lead to actionable decision for example on: new software development, user training, or marketing actions.

In this process, data collecting consists of phases 2-6.

B. Challenges of Automatic Usage Data Collecting

Fabijan et al. have described the challenges and limitations of customer feedback and data collection techniques in their literature review of software R&D [5]. The scope of their literature review included also manual and qualitative techniques such as interviews and observations, but the sources and challenges concerning automatic usage data collecting from the software product itself were as listed below.

- *Incident reports:* Available only after an incident.
- *Beta testing:* Only partially developed interfaces and functionality.
- *Operational and event data:* Security issues when such data is transmitted, potentially high amounts of data.
- *A/B testing:* Potentially confusing for customers when exposed to different versions.

Similarly, Sauvola et al. [6] described feedback gathering and its challenges as a part of software development companies' R&D efforts. Although their multiple-case study involved also many more feedback types than the automatically collected usage data, their descriptions of the cases implied various related challenges. We understood these as follows.

- *Permission checks.* The authors point out that in some specific domains the automated data collection from end-users is highly regulated and thus not executed at all.
- *Various sources of feedback.* Consolidating the feedback coming from various customers was seen as a challenge, and its processing relied heavily on its user's competence.
- *Only incident reports available.* Feedback was only gathered for troubleshooting purposes and not e.g. for improving existing products.
- *Systematic implementations are missing.* Although some mechanisms are in place to collect feedback and even product data, their implementations lack the systematic approach.
- *Difficulties to store, analyze, and integrate.* Even if feedback was gathered in most of the case companies, they reported having issues in storing, analyzing and integrating it back to the developers' processes.
- *Data availability and transparency.* The information about the collected data types as well as who and for what was it used was difficult to spread around the case companies. A reason for this, for example, was that the different parts of the development organization can see the data collecting as a risk as its use for new product development can cannibalize the current product markets.
- *Channels are not working.* As no systematic and organization-wide ways of feedback collecting were present, the feedback gathered in one place was regarded

useless although it could have been in high value in the next place.

III. EVALUATION CRITERIA FOR USAGE DATA COLLECTING APPROACHES

The challenges of usage data collecting are now extracted into evaluation criteria, which are fine-tuned based on the discussions with the case company. The challenges and limitations of usage data collecting can be consolidated as follows:

- *The amount of use cases for the collected data.* The number of use cases can be either too low or too high. Although there could be various uses for the same collected data, it might be used blindly to serve only a single purpose. On the other hand, the whole data collecting can face the critical challenge of trying to serve so many purposes and people that in the end it performs sufficiently to none of them.
- *The timeliness of the collected data.* Depending on the intended use, the timeliness of the data can form limitations to the collecting approach as well as to the source of data. For example, incident reports can be available only after incidents happen, and thus the use of such source has its natural challenges.
- *Continuous confusion for the users.* As mentioned, one of the well-known practices in the field of web development is A/B testing. Its implementation needs carefully planning, though. The more continuous the collecting is, the higher the risk of continuously introducing partially developed interfaces and functionality to users, who can find this troubling after a while. In addition, the collecting can affect the performance of the system.
- *Laws, regulations, and permissions.* Especially when the same data collecting approach is to be used in various different domains and countries, the related laws and regulations are going to be different for each situation. These checks for the data collection's legality take different amounts of time in each case thus enabling the data collecting in different cases at different times.
- *Privacy and security.* In addition to the overall permission checks, the security and privacy issues need to be addressed sufficiently by the collecting approach.
- *Various sources for the data collecting.* A high amount of sources creates a twofold challenge. The unification of the different types of data has to happen in a phase of its own (cf. phase 7 in Section II-A), or then the analyzer (cf. phase 9) has to have the capabilities to present and process the different types of data.
- *Lack of a systematic approach to collecting.* If a systematic approach is missing for the collecting, it is obvious that each phase of the data-driven software development is going to present new difficulties and challenges (e.g. difficulties to store, analyze, integrate etc.). These might be different in each case and they depend on the involved persons and their capabilities.
- *Availability, transparency, and usability of the collected data.* Even if the organization had first decided on what

kind of things they want to use the collected data, there are challenges also in how to make the data available, attractive, and usable for the right people. Thus, both the channels for distributing but also the tools for example for visualizing it (i.e. make the data usable) have to be sufficient enough for the selected audience.

We also analyzed these challenges from the perspective of the case company and used them as the basis of designing the evaluation criteria of usage data collecting approaches. The organization listed the challenges they felt were related to their case after discussing the overall topic of usage data collecting with us. Although each of the challenges they listed was found already from the list above, their descriptions of the challenges bring understanding to a more concrete level, which we try to emphasize with the examples linked to each criterion.

- *Timeliness*. When can the data be available? Does it have a support for real-time?
- *Targets*. Who should benefit from the data? What is the intended use? Does the approach support many targets? Does it produce different types of data or only one? "Do we want results for troubleshooting or for new product development?"
- *Effort level*. What kind of a work effort is needed from the developers to implement the approach. "How does the selected technology affect the production code? What is the work effort needed for the implementation?"
- *Overhead*. What kind of drawbacks are acceptable? "How does the collecting approach affect the implementation environment, e.g. downtime and performance?"
- *Sources*. What sources of data can be used? Does the approach support many source platforms? "Different kinds of technological environments – Where to focus our implementation efforts?"
- *Configurability*. How configurable the technological approach is? Can the collecting be switched on and off easily? Can it change between different types of data to collect? "Is the collecting easy to switch on and off? Is the approach producing data in the right level of details?"
- *Security*. Can the organization who developed the collecting technology be trusted with the collected data? Is the data automatically stored by the same organization?
- *Reuse*. How can the technology be reused? Is it always a one-time solution or can it be reused as it is straightforward with another target application?

IV. TECHNOLOGICAL APPROACHES FOR USAGE DATA COLLECTION

Next, we will go through a few technological approaches for the automated collecting of usage data. The abstract viewpoint is selected to not get stuck with the specific tools that happen to be around in 2016. Rather, the goal is to gain deeper understanding in how such tools and possible approaches work and how that is reflected in selecting them.

A. Manual Implementation

In the manual implementation the developer adds extra statements to the relevant locations of the software. On one hand, this highlights the flexibility of the approach – it does not limit the *timeliness*, *targets*, *sources*, or *security* in any way. On the other hand, adoption to new targets and sources would require significant rework making the *reuse* practically impossible. However, if additional functionalities such as run-time flags are added to the statements, switching the collecting on and off becomes significantly easier. This increasing *configurability* correspondingly increases the already high level of work *effort* needed for the implementation though. As a benefit, the approach almost guides the developer to collect data only from the intended sources, minimizing the *overhead* to the performance and of irrelevant data.

To conclude, there are only few real limitations with this approach. The needed work effort is high though, so e.g. if the code base is vast the approach can be come inappropriate. Therefore, we conclude that the approach is best suited either for the first few try outs with data collecting or for cases where the target and the source are particularly well focused.

B. Automatic Instrumenting with a Separate Tool

There are multiple tools, e.g. GEMS [7], that can automatically instrument the code for various data logging, quality assurance and performance monitoring purposes. This approach frees the programmers from the manual work and reduces the probability for errors lowering the *effort* significantly. Similarly, the *reuse* possibilities of the data collection should be high with automated tools since they are developed to work with different target applications in the first place.

However, the automatic tools are typically focused on only one type of *source* or *target*. Therefore, the *overhead* is likely to grow rapidly as the source cannot be set as specifically as with the manual approach. There are exceptions to this as well, and for example the framework presented in [8] balances its monitoring coverage with overhead automatically. Although these problems in general can be reduced by using highly *configurable* instrumentation tools when available, these criteria, along with *security and timeliness*, are almost completely intertwined with the specific tool selected. This highlights the inflexibility of the technological approach. The ideal case for this approach could be one with high importance in low implementation effort, such as a case with a huge code base, and with targets that need monitoring from the whole target application or even from many similarly developed target applications.

C. Aspect-Oriented Approach

Aspect-oriented approach is something of a mixture from the automatic instrumentation and the manual implementation. The research presented in [9] and [10] use aspect-oriented programming as a tool for code instrumentation. Further on, the use of aspect-oriented programming for usage data collection has been proposed in [11]. Additionally, in [12] the separation of similar monitoring code from the actual

system code is highlighted, which could perhaps respond to the challenge of various data sources.

One important benefit of aspect-orientation is its expressive power. While automatic instrumentation is typically triggered by entering (or leaving) a function, the aspects can include more complex conditions for executing the data collection code. Aspect-based instrumentation allows the instrumentation to be system and application specific, which focus the collecting better on the relevant *targets*. This should also lead to optimized balance between the additional *overhead* and quality of the data.

The expressive power of AOP makes the approach similar to the manual implementation in its flexibility to create solutions that can be optimized by their *timeliness*, *configurability*, and *security* to suit any situation. On the other hand, the work *effort* needed for the implementation is not as high since the instrumentation is automated. However, learning to use AOP surely takes its toll if the developer is not familiar with the paradigm otherwise.

From the perspective of *reuse*, the aspect-oriented approach has both its limitations as well as benefits. If the different target applications are developed in such a similar fashion that the targeted data collecting places use the same syntax, the reuse should be very straightforward. Obviously, this sets a strict limitation to the reuse. On a more general level, the approach is depended on an available AOP library for the specific target application’s programming language. If the language changes between the *sources*, i.e. the target applications, the reuse becomes much more difficult. This approach suits particularly well cases which need the same kind of system wide monitoring as the tool instrumentation’s ideal case, but which at the same time require more flexibility from the collecting. An available AOP library for the case’s programming language is obviously a critical limitation.

D. Alternative Implementation of a UI Library

An alternative implementation of a user-interface (UI) library can be set to automatically collect usage data. Because the user interaction is usually implemented with standard UI libraries, their components can be altered so that they include the collection of usage data within them. Similarly to automatic instrumentation, this approach frees the developers from the repetitive implementation *efforts*. Correspondingly, the issues are similar as well – data is easily collected also from unnecessary *sources* causing extra performance *overhead* and difficulties to the analysis phase. Although usage data is mainly linked to the UI and the types of data a UI library includes, some *targets* need integration with data types that are beyond the reach from these altered UI libraries.

On a more positive note, the approach has no limitations to the *security* or *timeliness*, and the *configurability* can be increased much like in the manual approach. As a matter of fact, this can be even easier if a differently altered UI library is deployed according to each requirements of a new case. The *reuse* of the implementations with this approach can be

TABLE I
SUMMARY OF THE TECHNOLOGICAL APPROACH EVALUATIONS.

Criteria	Technologies				
	Man.ins.	Tool ins.	AOP	UI	E.E.
Timeliness	+	-	+	+	-
Targets	+	-	+	-	-
Effort	-	+	+	+	+
Overhead	+	-	+	-	-
Sources	+	-	-	-	-
Config.	+	-	+	+	-
Security	+	-	+	+	-
Reuse	-	+	-	-	+

extremely easy, but new versions of the standard UI libraries create great issues as well.

E. Execution Environment

The data collection can also be done by the environment without any modification to the application. For languages like Java and JavaScript the virtual machine is an execution environment where method and function calls can be monitored by instrumenting critical places. One example of such systems is Patina [13] where the user input like cursor movements and key-presses are monitored. In these approaches, the implementation *effort* is often low, but the produced data requires a lot of post processing and there may significant performance penalties since it will reduce possibilities for advanced just-in-time compilation.

This approach often has a limited set *sources* and *targets*, but within that limited scope *reuse* is good. Similar to the automatic instrumentation tools, the *configurability*, *security*, and *timeliness* of the approach are intertwined heavily with the specific implementation, i.e. tool, that is implemented.

V. SELECTION FRAMEWORK FOR AUTOMATED USAGE DATA COLLECTING TECHNOLOGIES

We have summarized the evaluations of the technological approaches into the basis of the selection framework, i.e. Table I, giving each approach either a plus if it has a positive impact or if it does not restrict the implementation. An approach is marked with a minus sign if it limits the selection or the use of a data collecting implementation according to each criteria.

The first thing to do when selecting a technological approach to usage data collecting, is to rapidly **explore the case** to get a grasp of the most critical limitations to the technological approaches. These include things such as the size of the code base, availability of automated tools and AOP libraries for the target application’s language and platform, and access to the UI libraries and execution environments.

If any critical limitations are faced, the next step is to **reject the unsuitable approaches** accordingly. For example, if there are many security issues related to the data being collected or if data needs to be sent in real-time, the 3rd party tools used in approaches B and E might have critical limitations that cannot be avoided.

The following step is to **prioritize the evaluation criteria**. In addition to the explored case information, one should find

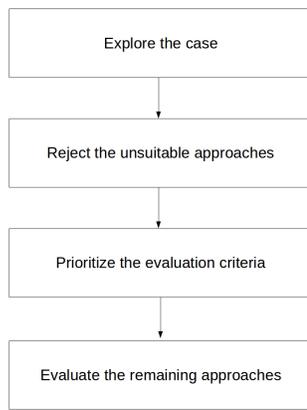


Fig. 2. Selection Framework for Technological Approaches.

out the goals different stakeholders have for the usage data collecting as these can have a major impact on the approach selection. If the goals are clearly stated, and the aim is e.g. to simply find out which of two buttons is used the most, manual instrumentation can work sufficiently. However, if the goal is stated anything like "to get an overall view of how the system is used" or if the goal is not stated at all, the more automated and more configurable approaches most likely become more appealing. Therefore, one of the most crucial things to find out in this step is to understand what different stakeholders want to accomplish with the collected data.

After this, the final step is to **evaluate the remaining approaches**. The plus and minus signs used in Table I work as guidelines in this, but their emphasis obviously varies on a case to case basis. To summarize, the selection framework is illustrated in Figure 2.

The further evaluation of how the selection framework performs is clear choice for future work. The setting with the case company is interesting for them, but it is attractive also academically as it provides an environment to study the "full-stack" that will be needed for the whole data-driven software development process in the end.

VI. CONCLUSIONS

In this paper, the main research problem was *how to select the right technological approach for automated collecting of usage data*. Literature reviews were performed to gain understanding of the context of the collecting processes and its challenges. These helped us form evaluation criteria for the technological approaches. We then described different approaches and evaluated them with the aforementioned criteria, which were then refined into the selection framework.

To summarize, the main contributions of this paper included literature reviews of the data-driven software development and of the challenges of collecting usage data, forming the evaluation criteria based on the studied challenges, description and evaluation of different technological approaches, and designing the selection framework for the technological approaches. The selecting of data collecting technologies is not

a straightforward challenge but it needs to be addressed each time an organization wants to start the data-driven software development. Obviously, various contemporary tool evaluations are available both in academic literature and especially in practitioner publications, e.g. blogs. However, the abstract perspective of this study to the technological approaches rather than today's tools should be valuable also in the long run.

Finally, the criteria described in this paper gives practitioners a good basis for the evaluation, but it works only as a guideline and case specific variations account heavily on the actual decisions. However, it provides them with a well-considered starting point in their journey towards ever more data-driven decision making.

ACKNOWLEDGMENT

This work is supported by Tekes (<http://www.tekes.fi/>) and Digile's Need for Speed program (<http://www.n4s.fi/en/>).

REFERENCES

- [1] E. Ries, *The lean startup: How today's entrepreneurs use continuous innovation to create radically successful businesses*. Crown Books, 2011.
- [2] H. H. Olsson, H. Alahyari, and J. Bosch, "Climbing the" stairway to heaven"—a multiple-case study exploring barriers in the transition from agile development towards continuous deployment of software," in *Software Engineering and Advanced Applications (SEAA), 2012 38th EUROMICRO Conference on*. IEEE, 2012, pp. 392–399.
- [3] H. H. Olsson and J. Bosch, "From opinions to data-driven software r&d: A multi-case study on how to close the 'open loop' problem," in *Software Engineering and Advanced Applications (SEAA), 2014 40th EUROMICRO Conference on*. IEEE, 2014, pp. 9–16.
- [4] R. P. Buse and T. Zimmermann, "Information needs for software development analytics," in *Proceedings of the 34th international conference on software engineering*. IEEE Press, 2012, pp. 987–996.
- [5] A. Fabijan, H. H. Olsson, and J. Bosch, "Customer feedback and data collection techniques in software r&d: a literature review," in *Software Business*. Springer, 2015, pp. 139–153.
- [6] T. Sauvola, L. E. Lwakatere, T. Karvonen, P. Kuvaja, H. H. Olsson, J. Bosch, and M. Oivo, "Towards customer-centric software development: A multiple-case study," in *Software Engineering and Advanced Applications (SEAA), 2015 41st Euromicro Conference on*. IEEE, 2015, pp. 9–17.
- [7] P. K. Chittimalli and V. Shah, "GEMS: A Generic Model Based Source Code Instrumentation Framework," in *Proceedings of the Fifth IEEE International Conference on Software Testing, Verification and Validation*. IEEE Computer Society, 2012, pp. 909–914.
- [8] J. Ehlers and W. Hasselbring, "A self-adaptive monitoring framework for component-based software systems," in *Software Architecture*. Springer, 2011, pp. 278–286.
- [9] W. Chen, A. Wassyng, and T. Maibaum, "Combining static and dynamic impact analysis for large-scale enterprise systems," in *Product-Focused Software Process Improvement*. Springer, 2014, pp. 224–238.
- [10] A. Chawla and A. Orso, "A generic instrumentation framework for collecting dynamic information," *SIGSOFT Softw. Eng. Notes*, vol. 29, no. 5, pp. 1–4, Sep. 2004. [Online]. Available: <http://doi.acm.org/10.1145/1022494.1022533>
- [11] S. Suonsyrjä and T. Mikkonen, "Designing an unobtrusive analytics framework for monitoring java applications," in *Software Measurement*. Springer, 2015, pp. 160–175.
- [12] M. Vierhauser, R. Rabiser, P. Grünbacher, K. Seyerlehner, S. Wallner, and H. Zeisel, "Reminds: A flexible runtime monitoring framework for systems of systems," *Journal of Systems and Software*, vol. 112, pp. 123–136, 2016.
- [13] J. Matejka, T. Grossman, and G. Fitzmaurice, "Patina: Dynamic heatmaps for visualizing application usage," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, ser. CHI '13. New York, NY, USA: ACM, 2013, pp. 3227–3236. [Online]. Available: <http://doi.acm.org/10.1145/2470654.2466442>

Deep Data Analyzing Method Based on Scale Space Theory

Ye Hu

College of Mechanical Engineering
Ningbo Dahongying University
Ningbo, China
Tongji University, Shanghai, China
pxhuye@126.com

Guangqun Chen

Ningbo Dahongying University
Ningbo, China
College of Mechanical
Engineering
Guangqunchen2016@126.com

Kun Gao

Zhejiang Business Technology
Institute, China
Zhejiang Wanli University
Ningbo, China
kungao@live.com

Abstract—Scale space theory has been introduced into the field of big data, but its research is still not deep enough and perfect because of the lack of universal theory and method. With deepening of big data processing applications, the research becomes more and more urgent. In view of the above question, this paper studies pervasive multi-scale data analysis theory and method. On one hand, we give the definition and partition of data scale as well as the relationship of multi-scale data set between the upper scale and lower scale based on concept hierarchy theory. On the other hand, we clarify the definition of multi-scale data analysis, study essence and classification method. Previous studies show that the proposed method has high coverage rate, high accuracy rate, lower error rate of support estimation degree and greater improvement the efficiency than the traditional algorithm.

Keywords: *Multi-Scale Space; Big Data; Frequent Item-set; Association Rules;*

I. INTRODUCTION

Broadly speaking, the scale is research object of units or measurement tools [1-3]. According to the data as the research object of the data analysis, scale is also a measurement unit of data [4-6]. And statistical measurement scales to variables as benchmark data for similar and data analysis of scale measurement based quasi should be the intrinsic properties of the data [7-9]. When researchers have a survey on the category data often correspond to data in the category of a property set, the attribute set can usually form a partial order structure explicit in the concept hierarchy, according to the concept hierarchy related concept is used to divide the data can be formed with the multi-scale characteristic of the data set [10-12].

From the present study, analysis and research of multi scale data is not fully expanded, the data types involved are more concentrated in spatial data, which limits the development space of multi scale analysis data [13-15]. In order to expand the theory of multi scale analysis research scope and depth in the field of data, this article from the following 3 aspects:

- 1) will the basic theory of multi-scale science into the general data set, put forward a more universal theory of multi-scale data;
- 2) describes the definition of multi-scale analysis of the data, essence and classification;
- 3) proposed the multi-scale data analysis algorithm frame and gives its theoretical basis, then analysis the algorithm framework is applied to association rules further, we propose a

multi-scale analysis of association rules algorithm, multi-scale data derivation of implicit association rules, the single scale data sets through knowledge push and push down, switch to other scales Level of knowledge, an analysis can be obtained multi-level knowledge. And the accuracy of the algorithm theoretically evaluated; finally, the real data set for experiment to prove the feasibility and efficiency of the algorithm.

II. MULTI-SCALES FOR DATA

A. Concept Hierarchy

Definition 1: concept hierarchy

Concept hierarchy C is a partial order set (C, \preceq) , where C is a finite set of concepts, \preceq denote C contains a partial order relation between concepts.

A category attribute data sets that have a distinct concept hierarchy partial ordering: every property $c_i(i=1, \dots, n)$ in attribute set $C = \{c_1, \dots, c_i, \dots, c_n\}$ can be used as a concept in limited concept set. Corresponding to the partial order relationship in limited concept set, attributes can form a partial order relation depend on domain knowledge. A attribute $c_i \in C$ is corresponding to a number of specific attribute values after the instances, denoted as $V_{c_i} = \{v_1, v_2, \dots, v_m\}$ (v_j present a specific discrete values or continuous intervals). Highly abstract of those attribute values in semantic formed attributes concept h_i , called as attribute values $v_j \in V_{h_i}$ ($j = 1, \dots, m$), belong to semantically h_i , denoted as: $V_j \in V_{c_i}^s$. In practical applications, the geographical scope of attributes concept set may be formed concept hierarchy: $(C_{\text{location}}, \preceq) = \{\text{county} \preceq \text{township} \preceq \text{village} \preceq \text{province} \preceq \text{city}\}$; Time Category attribute set can also form hierarchical concept: $(C_{\text{time}}, \preceq) = \{\text{day} \preceq \text{month} \preceq \text{years}\}$.

B. The Basic Concept of Data Scale

In the concept hierarchy (C, \preceq) , the partial order \preceq indicates concept involved range, the relative size of particle size, or the relative length of time amplitude, then call this concept hierarchy (C, \preceq) with a multi-scale feature. For example, the concept hierarchy $(C_{\text{location}}, \preceq) = \{\text{county} \preceq \text{township} \preceq \text{village} \preceq \text{province} \preceq \text{city}\}$ represents geographical area range from small to large; The concept $(C_{\text{time}}, \preceq) = \{\text{day} \preceq \text{month} \preceq \text{years}\}$ represents time amplitude from short to long. All of these concepts have the ability to represent the relative size of the data granularity, so they are all equipped with multi-scale characteristics. Using the concept of hierarchical multi-scale

features as the standard for dividing, the data set may be formed multi-scale data sets.

Definition 2: Data scale division

The attribute set of a certain category of data set DS forms a concept hierarchy with multi-scale characteristics: $(C, \preceq) = \{c_1 \preceq \dots c_i \preceq \dots \preceq c_n\}$, the set of attribute values of a certain concept $c_i (i=1, \dots, n)$ in the finite concept set C is $V_{c_i} = \{v_1^i, \dots, v_{m_i}^i\}$; According to the different attribute value $v_j^i (j = 1, \dots, m_i)$ of the concept c_i divide DS with the same attribute values to form an independent sub-data set, denoted as $ds_{c_i-v_j^i}$; Data set DS is divided into m_i sub data sets, which form a set of data with the concept of c_i as the partition granularity. This process is called the data scale division based on the concept c_i .

In the process of data scaling, the essential of scale is a unit of measurement with a certain semantic meaning, and its semantics is related to the concept of the concept hierarchy with multi-scale characteristics.

Definition 3: Data Scale

The concept c_i that is based on the classification of data set DS is the data scale of the data set in this division, denoted as $Scale_{DataSet=c_i}$.

Definition 4: Benchmark Scale Data Set

In the results of division for data set DS based on the concept c_i of concept hierarchy (C, \preceq) , all sub data set $ds_{c_i-v_j^i}$ is the meta scale data set under the data scale $Scale_{DataSet=c_i}$ for data set DS. If another scale data sets can be obtained through merging or decomposing from the meta scale data set, then the metascale data set is called the benchmark scale data set, the concept c_i corresponding to the benchmark scale data set is benchmark scale, denoted as Benchmark-Scale.

In general, all meta scale data set of a data scale $Scale_{DataSet=c_i}$ can be denoted as the $dataset_{c_i}$ to represent the data partition of the scale by the granularity is c_i . Using different concepts in the concept hierarchy as data scale to divide the same data set in multi-scale and multi granularity can obtain multi resolution and multi-scale data set. To investigate data in the manner of multi-scale and multi granularity can obtain implicit knowledge under different scales and different levels, so as to achieve the purpose of analyzing data from multiple scales. The proposed related concepts of data scale and multi-scale data set are similar to the data cube [16] in the representation. However, the multi-scale data and data cube have different focus in nature. The former concerns multi-scale characteristics of the data set itself, while the latter concerns the aggregation values.

C. The relationship between multi scale data sets

Definition 5: upper scale data sets and lower scale data sets

The data set DataSet is divided by data scale $Scale_{DataSet=c_x}$ and $Scale'_{DataSet=c_y}$, the obtained data sets are $dataset_{c_x}$ and $dataset_{c_y}$. If $c_x \preceq c_y$, then all meta scale data set $dataset_{c_y}$ under data scale $Scale'_{DataSet=c_y}$ are upper scale data set for all meta

scale data set $dataset_{c_x}$ under data scale $Scale_{DataSet=c_x}$, all meta scale data set $dataset_{c_x}$ under data scale $Scale'_{DataSet=c_x}$ are lower scale data set for all meta scale data set $dataset_{c_y}$ under data scale $Scale'_{DataSet=c_y}$.

The upper and lower scale data sets are relative concepts. The concept level of data scale in upper scale data set is higher, partition granularity is larger, data set containing data meaning is more generalization; The concept level of data scale in lower scale data set is lower, partition granularity is smaller, so it contains the meaning of data more clear and specific. The upper scale data sets clearly contain more data than the lower scale data set, so the upper scale data set is large scale data sets than the lower scale data set; correspondingly, lower scale data set is small scale data set than the upper scale data set. Large scale data sets and small scale data sets corresponding to the data scales are called large scale and small scale, but also for the relative concept. If the data scale $scale_{dataset=c_x}$ and $scale'_{dataset=c_y}$ in concept hierarchy $c_x \preceq c_y$, then c_x is a small scale, c_y is large scale, denoted as $scale_{dataset} \preceq scale'_{dataset}$.

III. MULTI-SCALE DATA ANALYSIS

A. Definition of Multi-scale Data Analysis

The main task of multi-scale data analysis has two aspects, that is, the multi scale implementation of data and knowledge of multi scale mining. The former belongs to the data preprocessing, the data partition of the scale can be realized; the latter needs improvement technique in detail, in the form of multiple scale data knowledge discovery, analysis and knowledge derivation are connected with each other. The definition of multi-scale data analysis is given in this paper.

Definition 6: Multi scale data analysis

Multi-scale data analysis refers to the data were multi-scale processing, form a set of multi-scale data, use, or improved data analysis techniques explore multi-scale data of implicit knowledge and analysis of, derived data of various scales form behind the knowledge between relations.

B. The essence of multi-scale data analysis

Scaling is the core content of multi-scale science research, literature [17] has needle of spatial data analysis put forward three kinds of scale transformation of the way, summed up includes two aspects, namely multi-scale multi scale conversion of data and knowledge conversion. In fact, the research method of multi-scale spatial data analysis is also applicable to the analysis of multi scale data. Is not difficult to see that the conversion of data at multiple scales of the principle is simple, but requires multi-scale forms of data were analyzed, workload is big; knowledge of scaling the workload is small, only a single scale of data analysis, but need to solve the problem of scale effect, complexity theory. Scale effect is the conclusion that a certain scale cannot be applied to another scale [18], that is, when the study object to change its scope of performance, size or amplitude, the analysis results will be changed with [19,20]. Similarly, in a certain scale data focuses on knowledge cannot be no difference in applicable in other scales in the data, we must use domain knowledge or multi-scale data set between relationship of analysis results are deduced, naturalization, is it possible to achieve true knowledge of multi-scale conversion. It

is obvious that the multi-scale transformation of knowledge is the essence of multi-scale data analysis. The definition of knowledge scale transformation is given below.

Definition 7: knowledge scale conversion

Let $SCALE$, $SCALE'$ are data scale of data set DS . The process of knowledge transfers from the $SCALE$ to $SCALE'$ is called knowledge scale transformation, denoted as $SCALE \xrightarrow{knowledge} SCALE'$. If $SCALE$ is large than $SCALE'$ $SCALE \leq SCALE'$, then the knowledge scale conversion from $SCALE$ to $SCALE'$ is called as scale down, denoted as $SCALE \rightarrow \downarrow SCALE'$; on the contrary, the knowledge scale conversion from $SCALE'$ to $SCALE$ is called as scale up, denoted as $SCALE' \rightarrow \uparrow SCALE$.

Due to data collected in the actual application, usually only appears to single scale, and there is no need to analysis data according to multiscale expression form. So in this paper, we will take the multi scale transform of knowledge as the research essence of multi-scale data analysis: research methods or algorithms. Knowledge of a single scale data set is derived and calculated, which is suitable for the knowledge of other scale data sets within a certain error range [21,22].

C. Classification of Multi Scale Data Analysis Algorithms

Based on the knowledge of scale transformation of the definition and study field scale conversion classification [23,24], from knowledge of the multi-scale conversion direction angle, multi-scale data analysis algorithm scale on Pushover analysis algorithm and scale pushdown analysis algorithm:

1) scale up analysis algorithm: using scale data from the lower concentration of knowledge, and domain knowledge, lower scale data set between the relationship, derived upper scale data hidden knowledge, and not the upper scale data set for direct analysis. Scale push analysis algorithm is designed to use the data of the micro one-sided knowledge of the macro comprehensive knowledge of the data.

2) scales down analysis algorithm: using data from the upper scale concentration of knowledge, domain knowledge and, lower scale data set between the relationship, is the lower scale data hidden knowledge, and not lower scale data set for direct analysis. The scale down push analysis algorithm is designed to utilize the data from the macroscopic and comprehensive knowledge of the details of the data to derive the local knowledge.

IV. MULTI SCALE DATA ANALYSIS ALGORITHM

A. Algorithm Framework

Based on the benchmark scale and scale conversion mechanism, this paper proposes the basic framework of Multi-Scale Data Analyzing Algorithm. The basic idea of the framework is: first select benchmark scale, on benchmark data sets used in data analysis algorithm to get the results of the analysis; then for any other target scale so, through the use of scale conversion mechanism and the conversion methods Standard BS analysis results or knowledge inversion to target scale so, direct access to the target scale data set behind the implicit knowledge of the approximate results, rather wrong

target scale data set for direct analysis, the ultimate realization of multi-scale data analysis. The specific steps of the algorithm framework are as follows:

Algorithm 1: the framework of multi-scale data analysis algorithm

INPUT: data set with the characteristic of multi-scale

OUTPUT: results of data analysis on the target scale

1. Select the benchmark scale of data set;
2. Use data analysis algorithm on data set of benchmark scale;
3. While the target scale is not benchmark scale
4. If target scale \leq benchmark scale
5. Run scale down analysis algorithm: benchmark scale $\rightarrow \downarrow$ target scale
6. Else if benchmark scale \leq target scale
7. Run scale up analysis algorithm: benchmark scale $\rightarrow \uparrow$ target scale
8. Exit while all data have been executed

In the algorithm, if the target scale is smaller than the benchmark scale, call scale down algorithm analysis, conduct knowledge scale down conversion, through benchmark scale down analysis, and reverse conduct the results on the target scale; if the target scale is bigger than the benchmark scale, of the analysis show; if so scale is larger than the benchmark scale, call scale up algorithm analysis, conduct knowledge scale up conversion, through the benchmark scale analysis, and reverse conduct the results on the target scale; If the interested scale has been analyzed, the algorithm ends and returns to the analysis of the results of each scale.

V. CONCLUSION

In this paper, the basic idea of multi-scale science is introduced into the field of data analysis. Data classification and data scale concept hierarchy scale definition is proposed based on the relationship between the upper and lower scale data set between multi-scale data sets is given, laid the foundation for the theory of multi-scale data; the core of knowledge conversion based on multi-scale, given the definition and classification of multi-scale data analysis; proposed the multi-scale data analysis of algorithm framework, and gives the theoretical basis of knowledge transformation in the framework of multi-scale algorithm, and this algorithm is applied to multi-scale analysis of association rules, proposed the multi-scale analysis of association rules algorithm, numerical analysis results with benchmark data sets. And the benchmark data set for target scale data set weight that is the target scale data sets of association rules behind the realization of the cross scale derived knowledge, provides the possibility for multi-scale decision.

ACKNOWLEDGMENT

This research is supported in part by Zhejiang Provincial Natural Science Foundation of China (No. LY16F020012), Ningbo Key Laboratory of intelligent home appliances (No.

2016A22008), Key Research Center of Philosophy and Social Science of Zhejiang Province-Modern Port Service Industry and Creative Culture Research Center, Zhejiang Province Public Technology Research and Industrial Project (No. 2015C32124), Projects in Science and Technique of Ningbo Municipal (No. 2014C10047 and No. 2012B82003), National Natural Science Foundation of China (No.61300202) and Natural Science Foundation of Shanghai (No. 13ZR1452900). Kun Gao is the corresponding author.

REFERENCES

- [1] Felzenszwalb, P., McAllester, D., and Ramanan, D.: 'A discriminatively trained, multi-scale, deformable part model', in Editor (Ed.)^(Eds.): 'Book A discriminatively trained, multi-scale, deformable part model' (IEEE, 2008, edn.), pp. 1-8
- [2] Kolda, T.G., and Sun, J.: 'Scalable tensor decompositions for multi-aspect data mining', in Editor (Ed.)^(Eds.): 'Book Scalable tensor decompositions for multi-aspect data mining' (IEEE, 2008, edn.), pp. 363-372
- [3] Mierswa, I., Wurst, M., Klinkenberg, R., Scholz, M., and Euler, T.: 'Yale: Rapid prototyping for complex data mining tasks', in Editor (Ed.)^(Eds.): 'Book Yale: Rapid prototyping for complex data mining tasks' (ACM, 2006, edn.), pp. 935-940
- [4] Wang, Z., Simoncelli, E.P., and Bovik, A.C.: 'Multi-scale structural similarity for image quality assessment', in Editor (Ed.)^(Eds.): 'Book Multi-scale structural similarity for image quality assessment' (Ieee, 2003, edn.), pp. 1398-1402
- [5] Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., and Witten, I.H.: 'The WEKA data mining software: an update', ACM SIGKDD explorations newsletter, 2009, 11, (1), pp. 10-18
- [6] Riedel, E., Gibson, G., and Faloutsos, C.: 'Active storage for large-scale data mining and multimedia applications', in Editor (Ed.)^(Eds.): 'Book Active storage for large-scale data mining and multimedia applications' (Citeseer, 1998, edn.), pp. 62-73
- [7] Huan, T., Sivachenko, A.Y., Harrison, S.H., and Chen, J.Y.: 'ProteoLens: a visual analytic tool for multi-scale database-driven biological network data mining', BMC bioinformatics, 2008, 9, (Suppl 9), pp. S5
- [8] Eldawlatly, S., Jin, R., and Oweiss, K.G.: 'Identifying functional connectivity in large-scale neural ensemble recordings: a multi-scale data mining approach', Neural computation, 2009, 21, (2), pp. 450-477
- [9] Li, T., Li, Q., Zhu, S., and Ogihara, M.: 'A survey on wavelet applications in data mining', ACM SIGKDD Explorations Newsletter, 2002, 4, (2), pp. 49-68
- [10] Danon, L., Diaz-Guilera, A., Duch, J., and Arenas, A.: 'Comparing community structure identification', Journal of Statistical Mechanics: Theory and Experiment, 2005, 2005, (09), pp. P09008
- [11] Fu, T.-c.: 'A review on time series data mining', Engineering Applications of Artificial Intelligence, 2011, 24, (1), pp. 164-181
- [12] Pal, S.K., and Mitra, P.: 'Pattern recognition algorithms for data mining' (CRC press, 2004. 2004)
- [13] Tsoumakas, G., Katakis, I., and Vlahavas, I.: 'Mining multi-label data': 'Data mining and knowledge discovery handbook' (Springer, 2009), pp. 667-685
- [14] Mucha, P.J., Richardson, T., Macon, K., Porter, M.A., and Onnela, J.-P.: 'Community structure in time-dependent, multi-scale, and multiplex networks', science, 2010, 328, (5980), pp. 876-878
- [15] Han, J., Kamber, M., and Pei, J.: 'Data mining: concepts and techniques' (Elsevier, 2011. 2011)
- [16] X. Luo, Zheng Xu, J. Yu, and X. Chen. Building Association Link Network for Semantic Link on Web Resources. IEEE transactions on automation science and engineering, 2011, 8(3), 482-494.
- [17] C. Hu, Zheng Xu, et al. Semantic Link Network based Model for Organizing Multimedia Big Data. IEEE Transactions on Emerging Topics in Computing, 2014, 2(3), 376-387.
- [18] Zheng Xu et al. Semantic based representing and organizing surveillance big data using video structural description technology. The Journal of Systems and Software, 2015, 102, 217-225.
- [19] Zheng Xu et al. Knowle: a Semantic Link Network based System for Organizing Large Scale Online News Events. Future Generation Computer Systems, 2015, 43-44, 40-50.
- [20] Zheng Xu et al. Semantic Enhanced Cloud Environment for Surveillance Data Management using Video Structural Description. Computing, 98(1-2):35-54, 2016.
- [21] Zheng Xu et al. Crowdsourcing based Social Media Data Analysis of Urban Emergency Events. Multimedia Tools and Applications, 10.1007/s11042-015-2731-1.
- [22] C. Hu, Zheng Xu, et al. Video Structured Description Technology for the New Generation Video Surveillance System. Frontiers of Computer Science, 2015, 9(6): 980-989.
- [23] Zheng Xu et al. Crowdsourcing based Description of Urban Emergency Events using Social Media Big Data. IEEE Transactions on Cloud Computing, 10.1109/TCC.2016.2517638.
- [24] Zheng Xu et al. Generating Temporal Semantic Context of Concepts Using Web Search Engines. Journal of Network and Computer Applications, 2014, 43, 42-55.

DFIPS: Toward Distributed Flexible Intrusion Prevention System in Software Defined Network

Xuesong Jia*, Danni Ren*, Yitao Yang*[†], Huakang Li*[†], Guozi Sun*[†],

*College of Computer, [†]Institute of Computer Technology

Nanjing University of Posts and Telecommunications

Nanjing, China

1214043025@njupt.edu.cn, 1141909032@qq.com, {youngyt, huakanglee, sun}@njupt.edu.cn

Abstract—With the evolution of the innovative software defined network (SDN), security issues have been taken into consideration. Intrusion prevention system (IPS) has widely deployed as a crucial measure in traditional network architecture to protect network from malignity. In spite of good capability of protection, IPS is still complained in many aspects, such as fixed deployment, single-point-detection and low utilization rate. In this paper, we propose a distributed flexible intrusion prevention system in software defined network (DFIPS). Our proposed DFIPS has three main modules: a classifier, a detector pool and a control agent. The classifier is in charge of slicing traffic. The detector pool then generates several detector nodes for detecting. The control agent interacts with the classifier and the detector pool, as well as higher level SDN controller APPs and OpenFlow switches. DFIPS integrating with SDN controller can easily achieve good load balancing among DFIPSs without repetitive deployment. We evaluate the two forms of DFIPS interaction and latency to show the advantage of DFIPS. In future, we would implement a more comprehensive DFIPS emulation to prove feasibility. We believe that the proposed DFIPS will be adapted in real networks eventually.

Keywords—*Intrusion prevention system (IPS); Software defined network (SDN); OpenFlow*

I. INTRODUCTION

Software defined network (SDN) is an innovative technology that enables a drastic change in network design and management [1]. By separating the control plane from the data plane and consolidating the control plane, SDN realizes a logically centralized and physically distributed network framework. OpenFlow [2], the most successful application program interface (API) designed for SDN, embraces the paradigm of highly programmable switch infrastructures. It enables an optimal network routing that traffic forwarding is no longer restricted to particular command on individual network devices. A global view of network makes control more flexible and centralized [3].

With the evolution of this new technology, security issues in traditional network still exist in SDN, such as suffering from Distributed Denial of Service (DDoS) attack. Besides, the logical centralization of control logic makes it easier for attackers to paralyze the whole network once they infect the controllers in SDN. So, how to protect SDN from abnormal or unpermitted flow must be taken into consideration.

Traditionally, intrusion prevention system (IPS) [4] which usually deployed at the edge of the trusted internal network, serves as the vital gate in detecting and responding to anomalies. Despite its popularity and advantage, IPS is criticized in many aspects [5], such as local concern, single-point-detection, sub-optimal utilization rate and unbalanced computing resource consuming.

In light of the problems above, we explore a different design. Instead of a inflexible deployment, we propose a distributed flexible framework of IPS in software defined network. We call it DFIPS. DFIPS is embedded in SDN controllers as an application so that it can be deployed flexibly and can have a global view of network that is convenient to supervising. DFIPS has three main modules: a classifier, a detector pool and a control agent. The proposed DFIPS also supports load balancing to achieve a global optimal utilization rate. We use network slicing and coordination work among DFIPSs to enable distributed fine-grained intrusion detection and prevention.

To summarize, the contributions of this paper are shown as follows:

- We present the detailed architecture of our proposed DFIPS (Section II) which involves: (i) three main modules presentation (the classifier, the detector pool and the control agent), (ii) load balancing and distributed processing. Besides, we also present the work procedure and give some assumptions.
- We present the initial evaluation of our design in an emulated SDN environment. We present the two forms of DFIPS interaction and latency to show the superiority of our design (Section III).
- We present a variety of outstanding related work on IPS among different areas (Section IV), before concluding in Section V.

II. DESIGN AND ARCHITECTURE

In traditional network architecture, IPS runs as a stand-alone device and is interference in traffic forwarding. To have a global view of anomalous traffic and lower delay without decrease detecting capacity, we present IPS in this paper as an application running on the centralized controller. This means that DFIPS has access to the network: all the communication between SDN controller and switches would pass through

DFIPS; and DFIPS could inspect all data wanting to pass through switches.

Now, we present the detailed description of each portion in DFIPS. The structure of DFIPS is shown in Figure 1. Our proposed DFIPS has two three portions: a classifier, a detector pool and a control agent.

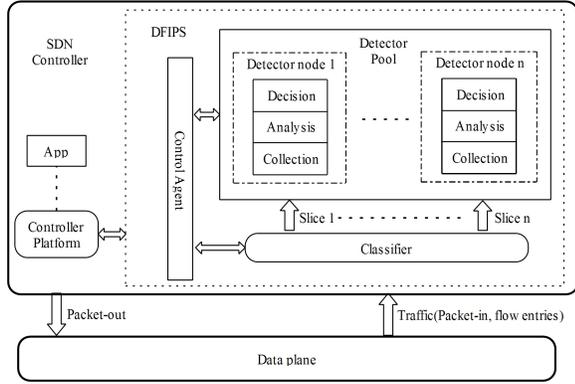


Fig. 1: The structure of DFIPS.

A. Main Modules

1) *Classifier*: The classifier is in charge of slicing traffic to improve resource allocation and realize a fine-grained distributed detection. We propose a modified FlowVisor [6] serving as a transparent proxy between switches and detector pool to implement network slicing. In our demonstration, a slice can be defined as (switch port, IP protocol, src/dst UDP/TCP port). Each slice is routed to the corresponding detector node independently.

2) *Detection Pool*: The detector pool generates a several detector nodes according to the number of slices from the classifier that each detector node has three submodule: collection submodule, analysis submodule and decision submodule. The detector node would be discarded when the slice is no longer exist.

Collection submodule keeps a record of messages that switches send to the controller. For further analyzing and queries, these records will be stored in the local database. Analysis submodule is for data analysis to find if flows are malicious, suspicious or normal via detection methods. Decision submodule has the authority to generate appropriate strategies to switches according to the analysis above. It sends information feedback to the control agent.

We propose a local database to store logs of every detection process and a remote database to share the feature library of abnormality collected from each single DFIPS in an associated region. In future, we can do data mining with semantic based algorithm [7]. Particularly, we tend to use R-trees [8] to access the feature library in remote shared database to make queries efficiently.

3) *Control Agent*: The control agent interacts with the classifier and the detector pool, as well as higher level SDN controller APPs and OpenFlow switches [9]. The control agent

also monitors and controls the running state of the classifier and the detector pool.

B. Load Balancing and Distributed Processing

There are two cases of load balancing and distributed processing as follow:

One DFIPS faces multi-switches. A single detector in DFIPS cannot deal with massive traffic from different switches in a very short period of time. Splitting traffic into several slices can enhance the parallel processing capabilities [10]. Each slice has a unified feature, such as a particular protocol, a common port, source/destination addresses. After slicing, a number n of slices are generated and then transmitted into the detector pool where spawns a number n of detector nodes. Besides, DFIPS will analyze the relevance of suspicious traffic among different switches to achieve better intrusion prevention performance.

Multi-DFIPSS face one switch. After a switch registering to several controllers, a master controller will be chosen. Others will be slave controllers. The DFIPS running on a master controller is the master DFIPS and in active mode. The rest of DFIPSS connecting to the same switch will be in standby mode. Once the master controller is down or congestion occurs in DFIPS, slave controller will take charge and DFIPS on it will be active. Furthermore, the DFIPSS in different locations exchange their detection result to update their local feature library of suspicious traffic in order to respond to a similar attack quickly.

C. Work Procedure

When traffic arrives at border switches, DFIPS control agent modifies the flow entries to make sure that every suspicious packet is forwarded to the detector pool for inspecting. The DFIPS control agent orchestrates actions of the detector pool and the command of upper level controller APPs. A detailed process of DFIPS is shown as follow (assuming the switches have been registered to SDN controllers):

- 1) Firstly, slicing the traffic into several slices and forwarding the slices to detector pool in where generates corresponding number of detector nodes.
- 2) Collection submodule in the detector node would store the digested information of each slice.
- 3) Then after the slice of traffic reach the analysis submodule, the analysis submodule would inspect it with several methods, such as high speed deep packet inspection (DPI) [11] and flow level detection [12].
- 4) Then the decision submodule would give suggestions responding to analysis results. If an abnormality is not detected, the DFIPS control agent would control the packet forwarding to upper level controller platform for normal process. Further, if detecting an abnormality and concluding that the packet is malicious, the DFIPS control agent would send drop command of malicious packet to the related OpenFlow switches; if not, a count of this packet would be recorded as suspicious traffic and increase at a time. DFIPS control agent would modify

the flow entries to make sure that all these suspicious traffic would be forwarded to DFIPS. While the count is higher than a given threshold value, decision submodule would suggest DFIPS control agent to do an appropriate QoS (e.g. rate limitation) when forwarding the packet; otherwise, the packet would be forwarded to upper level controller platform just like normal traffic.

- 5) DFIPS records the detection process into the local database and updates the shared remote database for future analyzing.

To prevent conflicting rules caused by misconfiguration, some security strategies must be taken. We tend to use FlowChecker [13] to accomplish this mission.

D. Assumptions

The proposed DFIPS is running as an application on centralized controllers. We assume that controllers are absolutely safe and unoccupied. In our architecture, DFIPS possesses the highest priority to conduct traffic control. It is easy to implement owing to abundant priorities (32768) available in OpenFlow. We also assume the packet loss rate and random noise are very low that can ignore.

III. EVALUATION

We build a test-bed (shown in Figure 2) using POX controller [14] and Mininet [15]. In our test-bed, a modified snort [16] is utilized to simulate a detector node and IPS. In this topology, **OFS 1** and **OFS 2** are OpenFlow-enabled switches.

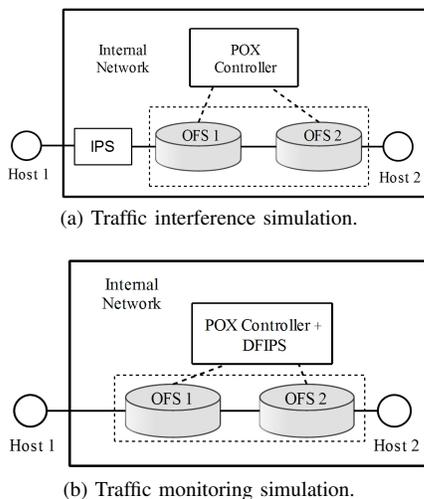


Fig. 2: Simulation diagram.

We first evaluate the two forms of DFIPS interaction to demonstrate the advantage of embedding DFIPS into SDN controller. In this simulation, we use ping to simulate the access request from the Internet to the internal network.

We ping twice (from Host 1 to Host 2) for each form of DFIPS interaction. The flow rules are not pre-configured. In traffic interference framework (Figure 2.(a)), the first ping time is 11.04ms, while the second ping time is 6.54ms. In traffic

monitoring framework (Figure 2.(b)), the first ping time is 10.86ms, while the second ping time is 0.862ms.

After the first ping, a flow rule is created and stored in the OpenFlow switch. This implies the second ping time does not have flow rules setup time. So the second ping time reduces largely. The second ping time in traffic monitoring framework is much smaller than in traffic interference framework implies that integrating DFIPS into SDN controllers has lower interference to normal communications.

Then, we evaluate the latency of slicing in DFIPS with topology Figure 2.(b). In this simulation, we examine the average new flow rules setup time on two situations: with slicing and without slicing.

The simulation shows that the average latency of new flow rules setup time with slicing (4.13ms) is more small than without slicing (7.25ms). It demonstrates that slicing traffic into several detector nodes has better performance.

IV. RELATED WORK

Now, we present some related work in intrusion prevention domain as below.

Recently, many proposed schemes have argued the novel IPS toward traditional network and software defined network. An SDN-based IPS deployment is proposed to support a unified scheduling of security applications in the whole network and load balancing among IPSs [17]. In this deployment, SDN controller manipulates IPSs in headquarters and branches to realize thread detection and load balancing between idle and busy IPSs. Different to DFIPS, this scheme cannot realize the flexible deployment.

Some work have shown the global, cooperative and distributed approaches of intrusion detection and prevention [18]. A cooperative IDS framework is addressed in [19] for cloud computing networks. In [20], an algorithm for combining observations from multiple vantage points is proposed. In [21], [22], a number of architectures that IPS/IDS works as a cluster are proposed. Take [21] as example, it proposes a general IDS architecture with splitting responsibility to other nodes by on-path distribution, replicating traffic to off-path nodes (e.g. IDS clusters) and aggregating result to split costly IDS processing. A plenty of previous work has focused on distributed security deployment. DEIDtect [23] is an elastic distributed intrusion detection framework that decouples the location of the protected network from IDS/IPS in the cloud and enterprise network. [24] shows the cooperation of active network management software and extensible hardware to stop an attack.

These proposed methods regard security facilities as middle-boxes [25]. And they seek to promote system performance on the basis of the original security hardware. On the contrary, our proposed DFIPS gets rid of the hardware security device and embeds into SDN controllers. This implies that our DFIPS can easily enable flexible deployment, global view and cooperation with other DFIPSs.

A SDN-based framework MalwareMonitor [26] is proposed to realize a comprehensive, distributed malware detection for

networks by tapping into outgoing traffic and detecting it with a number of detector nodes. Contrary to focusing on outgoing traffic, we concern incoming traffic from distrustful Internet to internal network. Besides, our proposed DFIPS is more flexible particularly in deployment.

V. CONCLUSIONS AND FUTURE WORK

In this paper, we present a distributed flexible intrusion prevention system in software defined network. The DFIPS we proposed decouples the location of security devices and protected points by integrating DFIPS into SDN controllers. DFIPS exploits the deployment of software defined network. One DFIPS can easily monitor several switches without repetitive deployment. This flexibility could reduce expenses largely on deployment. Further, DFIPSs in a certain region could commodiously achieve an optimal utilization rate by cooperating with each other. Slicing and virtualization enable the fine-grained detection and distributed processing.

We present the detailed description of design and implementation of each portion in the DFIPS. We evaluate the two forms of DFIPS interaction and the latency of slicing. The simulation result shows that DFIPS has lower delay and lower interference to normal communications.

In future, we would implement a more comprehensive emulation to solve other technical difficulties in DFIPS. For example, how to realize automatic network slicing and flexible detection nodes generation is a knotty problem. We also need to design a more accurate detection algorithm and a gentle controller coordination method. How to improve the performance of DFIPS is importance as well. Our ultimate goal is to get DFIPS widely deployed in real network environment.

ACKNOWLEDGMENTS

The authors would like to thank the reviewers for their detailed reviews and constructive comments. This work is supported by the National Natural Science Foundation of China (No. 61502247, No.61502243, No.11501302), the Natural Science Fund of province (BK20140895) and the Ministry of public security key experimental open project fund (No. 2015DSJSYS001).

REFERENCES

- [1] N. Feamster, J. Rexford and E. Zegura, "The Road to SDN: An intellectual history of programmable networks," In ACM SIGCOMM Computer Communication Review vol. 44, no. 2, pp. 87-98, 2014.
- [2] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner, "OpenFlow: Enabling innovation in campus networks," In ACM SIGCOMM Computer Communication Review, vol. 38, no. 2, pp. 69-74, 2008.
- [3] D. Kreutz, F. M. V. Ramos, P. Verissimo, C. E. Rothenberg, S. Azodolmoly and S. Uhlig, "Software-defined networking: A comprehensive survey," Proceedings of the IEEE, vol. 103, no. 1, 2015.
- [4] S. Vasanthi and S. Chandrasekar, "A study on network intrusion detection and prevention system current status and challenging issues," Advances in Recent Technologies in Communication and Computing (ARTCom 2011), 3rd International Conference on, pp. 181-183, 2011.
- [5] D. Stiawan, A. H. Abdullah and M. Y. Idris, "The trends of Intrusion Prevention System network," Education Technology and Computer (ICETC), 2010 2nd International Conference on, vol. 4, pp. 217-221, 2010.

- [6] R. Sherwood, M. Chan, A. Covington, G. Gibb, M. Flajslik, N. Handigol, T. Huang, P. Kazemian, M. Kobayashi, J. Naous, S. Seetharaman, D. Underhill, T. Yabe, K. Yap, Y. Yiakoumis, H. Zeng, G. Appenzeller, R. Johari, N. McKeown and G. Parulkar, "Carving research slices out of your production networks with OpenFlow," In ACM SIGCOMM Computer Communication Review, vol. 40, no. 1, pp. 129-130, 2010.
- [7] Zheng Xu et al. Knowle: a Semantic Link Network based System for Organizing Large Scale Online News Events. Future Generation Computer Systems, 43-44, 40-50, 2015.
- [8] A. Guttman, "R-trees: A dynamic index structure for spatial searching," In Proceedings of the 1984 ACM SIGMOD international conference on Management of data, pp. 47-57, 1984.
- [9] P. Bosshart, G. Gibb, H. Kim, G. Varghese, N. McKeown, M. Izzard, F. Mujica and M. Horowitz, "Forwarding metamorphosis: fast programmable match-action processing in hardware for SDN," In Proceedings of the ACM SIGCOMM 2013 conference on SIGCOMM, pp. 99-110, 2013.
- [10] S. Gutz, A. Story, C. Schlesinger and N. Foster, "Splendid isolation: A slice abstraction for software-defined networks," In Proceedings of the first workshop on Hot topics in software defined networks, pp. 79-84, 2012.
- [11] S. Kumar, S. Dharmapurikar, F. Yu, P. Crowley and J. Turner, "Algorithms to accelerate multiple regular expressions matching for deep packet inspection," In SIGCOMM '06 Proceedings of the 2006 conference on Applications, technologies, architectures, and protocols for computer communications, pp. 339-350, 2006.
- [12] Y. Gao, Z. Li and Y. Chen, "A DoS resilient flow-level intrusion detection approach for high-speed networks," In Proceedings of the 26th IEEE International Conference on Distributed Computing Systems, pp. 39, 2006.
- [13] A. Ehab and A. Saeed, "FlowChecker: configuration analysis and verification of federated OpenFlow infrastructures," In Proceedings of the 3rd ACM workshop on Assurable and usable security configuration, pp. 37-44, 2010.
- [14] <http://www.noxrepo.org/>.
- [15] <http://mininet.org/>
- [16] <https://www.snort.org/>.
- [17] L. Zhang, G. Shou, Y. Hu and Z. Guo, "Deployment of intrusion prevention system based on software defined networking," In Communication Technology (ICCT), 2013 15th IEEE International Conference on, pp. 26-31, 2013.
- [18] V. Sekar, R. Krishnaswamy, A. Gupta and M. K. Reiter, "Network-wide deployment of intrusion detection and prevention systems," In Proceedings of the 6th International Conference (Co-NEXT), Article No. 18, 2010.
- [19] C. Lo, C. Huang and J. Ku, "A cooperative intrusion detection system framework for cloud computing networks," In Parallel Processing Workshops (ICPPW), 2010 39th International Conference on, pp. 280-284, 2010.
- [20] A. Lakhina, M. Crovella and C. Diot, "Diagnosing network-wide traffic anomalies," ACM SIGCOMM Computer Communication Review, vol 34, no. 4, pp. 219-230, 2004.
- [21] V. Heorhiadi, M. K. Reiter and V. Sekar, "New opportunities for load balancing in network-wide intrusion detection systems," In Proceedings of the 8th international conference on Emerging networking experiments and technologies (CoNEXT), pp. 361-372, 2012.
- [22] F. Gadaud, "NIDS architecture for clusters," In Collaborative Technologies and Systems, 2005. Proceedings of the 2005 International Symposium on, pp. 78-83, 2005.
- [23] P. K. Shanmugam, N. D. Subramanyam, J. Breen, C. Roach and J. V. d. Merwe, "DEIDtect: Towards distributed elastic intrusion detection," In Proceedings of the 2014 ACM SIGCOMM workshop on Distributed cloud computing (DCC), pp. 17-24, 2014.
- [24] T. Sproull and J. Lockwood, "Distributed intrusion prevention in active and extensible networks," Proceedings of the 6th IFIP TC6 international working conference on Active networks, pp. 54-65, Springer-Verlag Berlin, Heidelberg 2007.
- [25] A. Gember, R. Grandl, J. Khalid and A. Akella, "Design and implementation of a framework for software-defined middlebox networking," In Proceedings of the ACM SIGCOMM 2013 conference on SIGCOMM, pp. 467-468, 2013.
- [26] Z. Abaid, M. Rezvani and S. Jha, "MalwareMonitor: An SDN-based framework for securing," In Proceedings of the 2014 CoNEXT on Student Workshop (CoNEXT Student Workshop), pp. 40-42, 2014.

Knowledge Flow-based Semantic Organization of Resources about Foreign Language Learning

Li Li

Shanghai University of Political Science and Law
Shanghai, China
e-mail: lily2211@126.com

Abstract—With the advent of the big data age, more and more resources about foreign language learning are put on the Web and open to the public, which brings trouble on the foreign language learners in acquiring these unfamiliar learning resources effectively and efficiently. To solve the problem, this paper proposes a method of organizing the learning resources in the form of knowledge flow network and provides learners with learning resources in the form of knowledge flow. Compared with other semantic organizations of resources, the proposed knowledge flow-based organization of resources not only helps the learners obtain the target resources properly but also displays them the study route related to the target resources. Experimental results indicate that the proposed method can improve the utilization rate of learning resources and the efficiency of Foreign Language Learning.

Keywords- Foreign Language Learning; semantic organization; recommendation system; knowledge flow; knowledge mining; e-learning.

I. INTRODUCTION

With the advent of the big data age, more and more resources about language learning are put on the Web and open to the public, such as course websites of universities, learning resources database, MOOCs, etc.. A foreign language learner can get the resources not only from his teachers but also from the Web which contains a large amount of high quality resources from the first language country. There is so much variety that it is difficult for a beginner of foreign language learner to obtain the appropriate resources from such massive online learning resources.

Many methods have been proposed to help a user find learning resources more easily and exactly. Search engines, such as Google, Bing, Baidu, etc., are the most popular tools. Search engines consider all learning resources on the Web as common webpages. Search engines crawl the learning resources from the Web, indexed them by terms, provide a search service based on keyword matching for the users. However, the search engine has trouble with search learning resources effectively for the beginner of foreign learning, which is caused by the implementation of search engine. The realistic scene is as following. When a learner searches a resource by a keyword, the search engine will feedback him a large amount of results containing the keyword, which may puzzle the user how to select an appropriate result from the returned list. To solve the problem of search engine,

recommendation systems are proposed to provide more exact results for the users based on some semantic computing technologies, such as, replacing keyword matching with similar computing, organizing the resources in semantic forms, and recommending the related resources based on association relations. Although recommendation systems work better than traditional search engines, they cannot still meet the requirement of the beginners of foreign language learning. For example, the recommended results are discrete, which is of no help in providing well-organized learning resource for the foreign language learners and improving the efficiency of language learning.

The organization of learning resources determines the final service pattern based on the resources. To the beginners of foreign language learning, they expect to get not only the target resource but also the knowledge relations between the target resource and other related learning resources. For example, what are basic knowledge points he must acquire before starting to learn this knowledge point? What is the next knowledge point he can study after finishing learning this knowledge point? We summarize the following shortcomings of the current organization structure which have to be confronted by the learning resources searching task.

- Lack of the proper sequence of knowledge resources. Both the linear index structure employed by search engines and the semantic index employed by recommendation system do not hold the knowledge sequence of learning resources
- Isolated islands of learning resources. There are some isolated islands in resources utilization rates of which are very low. When a resource is at the tail of index sequence, the possibility that will be accessible is very low.

Based on the above analysis, the knowledge flow is suitable to organize knowledge points of learning resources to support the above requirements. Therefore, this paper proposes a new semantic organization of learning resources, knowledge flow-based semantic organization structure, which organizes the learning resources in the form of knowledge flow network and provide learners with learning resources in the form of knowledge flow. The framework of knowledge flow-based service of learning resources is shown in Figure 1, the main components of which will be discussed in Section III and Section IV. To demonstrate that the proposed methods are

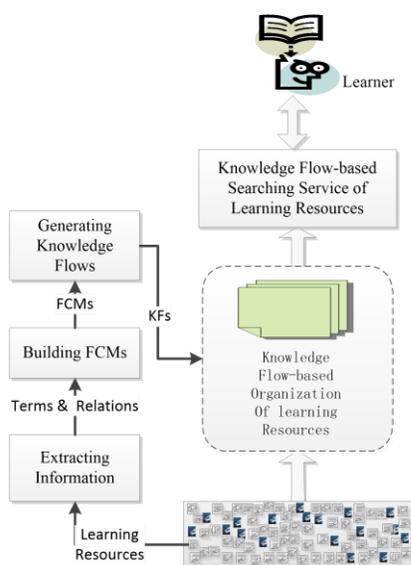


Figure 1. The framework of knowledge flow-based service of learning resources

effective, experiments are conducted in this paper and the results show that our approach works well in improving the utilization rate of learning resources and the efficiency of foreign language learning. The contributions of the paper are as follows:

- Propose a method of representing the learning resources based on fuzzy cognitive map, which is efficient for mining the semantic relations among learning resources.
- Propose a knowledge flow-based semantic organization structure to organize the learning resource, which can support new types of knowledge services.

The rest of this paper is organized as follows. In Section II, related work is discussed. In Section III, a method of representing the learning resources based on fuzzy cognitive map is proposed. In Section IV, a knowledge flow-based semantic organization structure is proposed to organize the learning resources. In Section V, experimental results are shown to verify the effectiveness of the proposed methods. Finally, Section VI concludes this paper and offers further research work.

II. RELATED WORK

A. Representation of Learning Resource

Most of learning resources on the Web are shown as documents[13], which can be presented by Vector Space Model (VSM), Fuzzy Cognitive Map (FCM), and so on.

Vector space model or term vector model is an algebraic model for representing text documents (and any objects, in general) as vectors of identifiers, such as, for example, index terms [1]. A definition of a document is that it is made of a joint membership of terms which have various patterns of occurrence [2]. These occurrence patterns are often disregarded because of complexity and single word statistics, i.e, document indexing, are used instead [3][5]. The IR pioneer Luhn used the

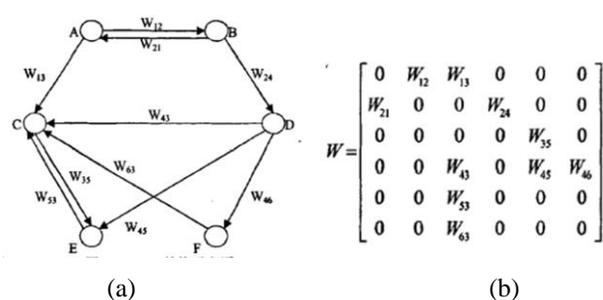


Figure 2. A sample of FCM and its adjacency matrix

term frequency as content descriptive features for documents [4] and it is still today the most used document description method, see e.g. [6][7].

A Fuzzy Cognitive Map (FCM) is a graphical model for knowledge representation and reasoning[8]. Knowledge representation is the basic ability of FCM. Compared with VSM, FCM consists of concepts and relations among concepts. When it is used to present learning resource, it is better than VSM in the aspect of presentation ability. Furthermore, FCM can represent not only causal relations between concepts but also knowledge of different granularity levels. Recent years, FCM has been improved in various aspects [9][10][11][12]. Elpiniki proposed a method to train a FCM, which is based on the nonlinear Hebbian learning and the differential evolution algorithm [8]. Wojciech proposed a novel parallel approach to the learning of FCM in order to deal with large maps due to its high computational complexity [15]. Mateou proposed two algorithms for a multilayer approach developed to expand the capabilities of FCM, one is ML-FCM (Multilayer FCM), and the other is EML-FCM (Enhanced Multilayer FCM) [14]. Luo et al. utilized the Hebbian Learning Rule and the Unbalance Degree to improve a FCM, so as to make it applicable to self-learning [16]. In this paper, FCM is used as the basic presentation model for Foreign Language Learning resources.

B. Semantic Organization of Learning Resources

Learning resources should be organized in some semantic forms in order to support high efficient searching and recommendation for Foreign Language Learning and Teaching. Because learning resources are usually documents, they can be organized by the same methods with those of Web content organization [24][25]. Semantic Web employs such techniques as URI, XML, RDF, OWL to identify, describe and associate the resources and settles the problem of semantic association of Web/learning resources. But because ontology can't be fully automatic, it is difficult to deal with the increasing number of resources and apply to semantic organization of massive learning resources. In [17], a resource space model is proposed based on semantic link network. [18][19][20] use semantic link network to organize resources efficiently. [19] proposed the construction method of association semantic link network. [18] builds a multi-layer association link network of keywords.[20] proposed the method to build the similar semantic link network on a large scale of web resources. In this paper, semantic link network is used as a basic tool to organize knowledge flows of learning resources.

III. REPRESENTATION OF LEARNING RESOURCE BASED ON FUZZY COGNITIVE MAP

Because most of learning resources are documents (webpages) or can be converted into documents, we use Fuzzy Cognitive Map (FCM) to represent a learning resource document. FCM is a graphical model which is comprised of concepts (nodes) and the relations between concepts (edges). Figure 2(a) shows the logic structure of FCM and Figure 2(b) shows the storage structure of FCM [22].

A document can also be described as a series of keywords and the relations among keywords. According to the structure of FCM, a learning resource is defined as:

Definition 1. Learning Resource (LR): A learning resource is formed by attributive keywords and the semantic relations among attributive keywords and denoted as

$$LR = \left\langle P, R^P \right\rangle = \left\langle \begin{matrix} P = \{t_k \mid t_k \in T, 0 \leq k \leq |T|\}, \\ R^P = \{(t_i, t_j, w) \mid t_i, t_j \in P, 1 \leq i, j \leq |P|, 0 \leq w \leq 1\} \end{matrix} \right\rangle, (1)$$

in which, T is the domain keywords list, P is the attribution set of Learning Resource LR and is formed of the keywords t_k that belongs to T . $|T|$ is the size of T . R^P is the set of semantic relations in P , each semantic relation is described by a triad (t_i, t_j, w) , in which w denotes the strength of the relation between keywords t_i and t_j . $|P|$ is the size of P .

Given a learning resource, to represent it in the form of definition 1, two major tasks should be fulfilled.

The first one is to extract concepts from the given learning resource. There are many researches on how to extract keywords/concepts from the document [21]. This paper just takes advantage of the existed methods to do the work.

The second one is to mine the semantic relations among keywords/concepts. It is difficult to mine the causal relations among the concepts based on a single document. This paper uses association relation to replace the causal relation among concept nodes, because the causal relation is a special association relation, easy to mine from document automatically. There are also many methods to mine the association relation, such as the Apriori Algorithm and all kinds of improved algorithms [23].

IV. ORGANIZATION OF LEARNING RESOURCES BASED ON KNOWLEDGE FLOW

To the learning of foreign language, the amount of learning resources is very large, on the contrary, the knowledge points of the language are much fewer than learning resources. Each learning resource consists of one or several knowledge points and can be mapped to the basic knowledge points. The knowledge flow can be used to organize both the basic knowledge points and the learning resources.

A. Knowledge Flow

Definition 2. Knowledge Flow (KF): Knowledge Flow is a linear directed graph whose vertices are knowledge points (or

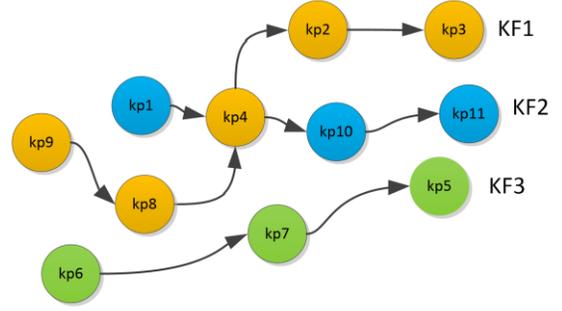


Figure 3. A sample of knowledge flow network

learning resources) and whose edges are the relations among knowledge points. All vertices of KF are listed in the order of $\{lr_1, lr_2, \dots, lr_n\}$ and $i = 1, 2, \dots, n-1$. The edge $\langle lr_i, lr_{i+1} \rangle$ means the knowledge point (or learning resources) lr_i should be learnt before lr_{i+1} .

A sample of knowledge flow network is shown in Figure 3. In Figure 3, the network consists of 11 knowledge points which are listed in Table I. For example, $\{kp1, kp4, kp10, kp11\}$ in Figure 3 is a knowledge flow with four knowledge points which indicates that the best learning route should be $kp1 \rightarrow kp4 \rightarrow kp10 \rightarrow kp11$.

The procedures of building the knowledge flows are as follows:

- Step 1: Mine knowledge points from learning resource. (This can be realized by concept extraction method.)
- Step 2: Verify the mined knowledge points by domain experts (in FLL they may be the teachers).
- Step 3: Mine the association relations among knowledge points (This can be realized by association relation mining method, such as Apriori.)
- Step 4: Connect the knowledge points by the association relations mined in step 3 and get the initial knowledge flows.
- Step 5: Verify the knowledge flows by domain experts (in FLL they may be the teachers).

When the knowledge flow of basic knowledge points is mined, the knowledge flow of learning resources can be obtained according to the maps from learning resources to knowledge points.

TABLE I. EXAMPLES OF KNOWLEDGE POINT IN FLL

No	Knowledge Point in FLL
kp1	word
kp2	the present tense
kp3	the future tense
kp4	the perfect tense
kp5	attributive clause
kp6	adverbial clause
kp7	object clause
...	...

B. Knowledge Flow Network

Definition 3. Knowledge Flow Network (KFN): Knowledge Flow Network is a set of knowledge points and their knowledge flows. KFN is denoted as

$$KFN = \{KP, KFS\}, \tag{2}$$

where KP is a set of knowledge points and KFS is a set of knowledge flows in KP . KFS is denoted as

$$KFS = \{kf_1, kf_2, \dots, kf_i, \dots, kf_n\}, \tag{3}$$

where kf_i is the i^{th} knowledge flow which is defined by Definition 2.

Knowledge flow network combines knowledge flows to support flexible usage of knowledge flows. For example, in Figure 3, three knowledge flows (shown in different colours) are combined into a network. Knowledge flows KF1 and KF2 have a joint knowledge point and connect together. When $kp4$ is checked, there are two available learning directions for the next step of learning.

The difference between knowledge flow network and knowledge point network is as follows. The paths in knowledge flow network are limited according to knowledge flows. For example, in Figure 3, when $kp8$ is the current point, the only path is $\{kp8, kp4, kp2, \dots\}$, that is to say that the path $\{kp8, kp4, kp10, \dots\}$ will not be recommended to the learner. Thanks to this limitation, the efficiency will be greatly increased, especially when the amount of points and knowledge flows are very large, for with the growth of joint points the amount of paths increases rapidly without the limitation.

V. EXPERIMENTS

In this section, two experiments are designed to validate the proposed methods. One is used to evaluate the accuracy of knowledge flows construction, and the other is to evaluate the effectiveness of the helpfulness of the proposed method in improving the foreign language learning.

A. Experiment on the Construction of Knowledge Flow Network

The knowledge flow of learning resources is the foundation of learning resource recommendation service, thus it is important to build a highly accurate knowledge flow based on the given learning resources. This experiment is designed to evaluate the accuracy of knowledge flows construction method. In this experiment, we select the College English Learning, which is a required course in Chinese university, as the learning task.

1) Experiment Process

a) *Build the learning resources set:* In this experiment, we collect all learning resources related to the knowledge points appearing in Book One of College English and build the learning resources with them. All kinds of learning resources used in the dataset are described in Table II, which consists of not only the source materials within college but also the materials from the Web.

TABLE II. LEARNING RESOURCE DATASET FOR COLLEGE ENGLISH LEARNING

#	Types of Learning Resources	Description
t1	Intensive reading in book	30 articles
t2	Exercises in book	1000 exercises
t3	Test paper in university	1000 exercises
t4	Words in book	2000 words
t5	Phrases in book	100 phrases
t6	Reading material on the Web	100 articles
t7	Test papers on the Web	500 articles

b) *Extract knowledge points from learning resource:* In this step, all materials in the learning resources set are converted to an unified form which is document in our method. As a result, the learning resource set is a document set in fact. Then, the necessary processes of text semantic mining are implemented on the document set, such as text extraction, word segmentation, keywords/terms extraction, and association rule mining. Next, the keywords/terms semantic network is constructed, whose nodes are keywords/terms and whose edges are assoction rules. Based on the keywords/terms semantic network, the concepts are extracted from the network according to the centrality of each node in the network. All the selected concepts form the candidate set of knowledge points.

c) *Optimize the mined knowledge points by domain experts :* Generally, the candidate set of knowledge points can be used as the knowledge points in the next step. However, improving the accuracy of knowledge points will do good to the accuracy of the final knowledge flow network. In this step, the college English teachers are employed as the domain experts to optimize the mined knowledge points. The optimization results are shown in Figure 4.

d) *Mine the association relations among knowledge points:* Each knowledge point is composed of a kernel term and its attributive terms. Each knowledge point can be seen as a transaction and each attributive term of the knowledge point can be seen as an item of the transaction. All knowledge points form the transaction set. As a result, we can use Apriori algorithm to mine the association rules in the transaction set. Furthermore, the association relations among knowledge

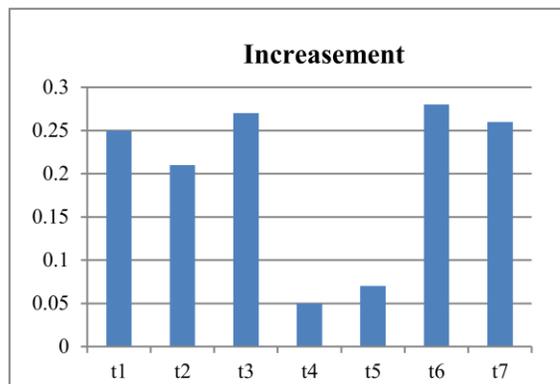


Figure 4. The increase of knowledge points after the optimization.

points can be mined by summarizing the association rules.

e) *Generate the initial knowledge flows:* Connect the knowledge points by the association relations mined in step d) and get the initial knowledge flows. To limit the amount of knowledge flows, a threshold of the weight of the association relation is used to control the association relation to join the construction of knowledge flows. The generated knowledge flows are influenced by the value of threshold, which is shown in Figure 5.

f) *Optimize the initial knowledge flows:* There are some defects in the automatic generated knowledge flows. For example, the lack of association rule will truncate a whole knowledge flow. The threshold used in step e) can also lead to the same question. Some error association rules can lead to the error knowledge flows. In the step, we also use college English teachers as the domain experts to optimize the initial knowledge flows. This step is more difficult than step c). The domain experts check each knowledge flow, revise the error connections in the knowledge flow, split the knowledge flow which is too long, and connect knowledge flows together that should be combined.

g) *Build knowledge flow network:* In this step, knowledge flows are connected by communal knowledge points in knowledge flows. As last, knowledge flows are connected together.

2) *Experiment Results*

Figure 4 shows the effectiveness of knowledge point extraction method. Figure 4 shows the increase of accuracy of knowledge points after they are optimized by domain experts. The results also reflect the accuracy of knowledge point extraction method. According to the figure, the accuracy of knowledge point extraction method is bigger than 0.7 in all seven types of learning resource set.

Figure 5 shows that the amount of knowledge flows is influenced by the threshold of association relation. Obviously, the amount of KFs decreases with the increase of threshold of association relation. When the threshold is bigger than 0.9, all association relations are filtered, which reduces KFs to zero. On the contrary, when the threshold is small, the amount of

KFs is big which will increase the complexity of the knowledge flow network and decrease the effective of the learning resources recommendation service. Therefore, the optimal threshold is expected which will be also our future work.

B. *Experiment on the effectiveness of Knowledge Flow-based Foreign Language Learning*

This experiment is designed to evaluate the effectiveness of Knowledge Flow-based Foreign Language Learning. Generally, the planned and ordered learning is more effective than the unplanned and disordered one. We group the participants into two groups. One group learns with the support of knowledge flow and the other group learns at random without the support of knowledge flow. The effectiveness of the proposed method can be got by comparing the scores of two groups.

1) *Experiment Process*

a) *Step 1:* Select 40 non-English major students from different departments in a university. All of the 40 students are divided into two groups randomly: one is experimental group, and the other is control group. Each group has 20 students.

b) *Step 2:* Define five learning tasks and each task consists of five knowledge points. Select five learning resources from the Web for each knowledge point.

c) *Step 3:* Design examination papers for each learning task.

d) *Step 4:* The students of control group will start their learning without any sequence. On the contrary, the students of experimental group start the learning according to the sequence of knowledge flows.

e) *Step 5:* The experiments are repeated on five learning tasks and all the students take three exams. Based on the exam results, the averages of students' scores are calculated and compared.

2) *Experiment Results*

The experimental results are shown in Figure 6. The results show that the experimental student groups concentrate more on knowledge level compared with those of control groups, which means that the knowledge flow-based organization and service

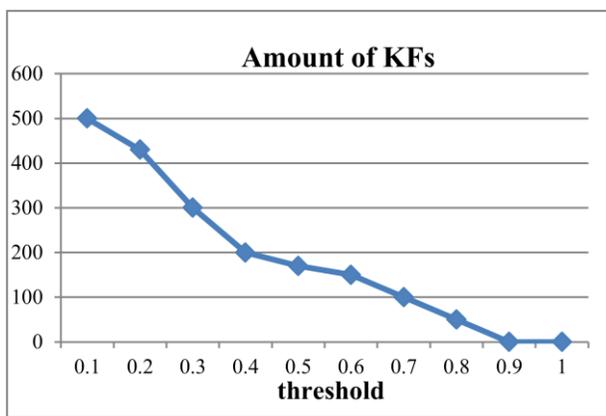


Figure 5. The amount of knowledge flows is influenced by the threshold of association relation.

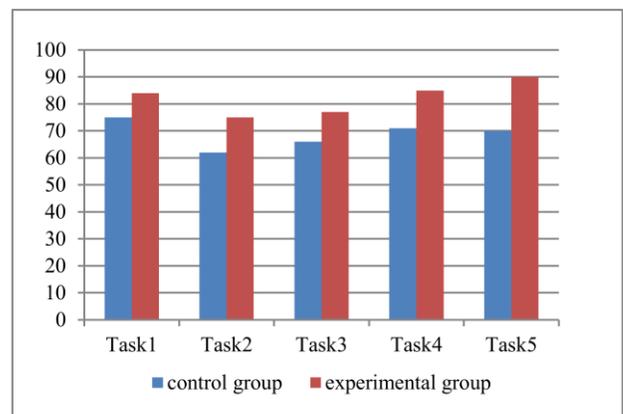


Figure 6. Knowledge flow network of knowledge points

of learning resources can improve the learning efficiency in Foreign Language Learning.

VI. CONCLUSIONS

To help the foreign language learners to find appropriate learning resources and improve the efficiency of Foreign Language Learning, this paper proposes a new semantic organization of learning resources, knowledge flow-based semantic organization structure, which organizes the learning resources in the form of knowledge flow network and provides learners with learning resources in the form of knowledge flow. Experimental results show that the proposed method works well in improving the utilization rate of learning resources and the efficient of Foreign Language Learning.

The future work is to design a systemic foreign language learning approach to bringing the proposed knowledge flow-based semantic organization structure into full play, so as to improve the efficiency of FLL.

ACKNOWLEDGEMENT

Research work reported in this paper was supported by the Science Foundation of Shanghai under grant no.16ZR1435500.

REFERENCES

- [1] G. Salton, A. Wong, and C. S. Yang, "A vector space model for automatic indexing," *Communications of the ACM*, vol.18, no.11, pp.613-620, 1975.
- [2] C. T. Meadow, D. H. Kraft, and B. R. Boyce, "Text information retrieval systems," Academic Press, 2007.
- [3] G. Salton, and C. Buckley, "Term-weighting approaches in automatic text retrieval," *Information Processing & Management*, vol.24, no.5, pp.513-523,1988.
- [4] H. P. Luhn, "The automatic creation of literature abstracts," *IBM Journal of Research & Development*, vol. 2, no.2, pp.159-165,1958.
- [5] R. R. Larson, "Introduction to information retrieval," *Journal of the American Society for Information Science & Technology*, vol.61, no.4, pp.852-853, 2010.
- [6] M. Zorman, V. Podgorelec, and P. Kokol, "Quest for the information: using intelligent search for finding telemedical sites," In *Proceedings of IEEE International Conference on Systems, Man, and Cybernetics*, pp. 4086-4091,1998.
- [7] N. Vljajic, and H. C. Card, "Categorizing Web pages using modified ART," In *Proceedings of Canadian conference on electrical and computer engineering*, pp.313-316,1998.
- [8] E. I. Papageorgiou, and P. P. Groumpos, "Two-stage learning algorithm for fuzzy cognitive maps," In *Proceedings of 2nd IEEE International Conference on Intelligent System*, vol.1, pp.82-87, 2004.
- [9] C. T. Lin, and C. S. G. Lee, "Neural-network-based fuzzy logic control and decision system," *IEEE Transactions on Computers*, vol.40, no.12, pp.1320-1336, 1991.
- [10] E. I. Papageorgiou, "Learning algorithms for fuzzy cognitive maps-a review study," *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, vol.42, no.2, pp.150-163, 2012.
- [11] G. Acampora, and V. Loia, "On the temporal granularity in fuzzy cognitive maps," *IEEE Transactions on Fuzzy Systems*, vol.19, no.6, pp. 1040-1057, 2011.
- [12] H. Song, C. Miao, W. Roel, Z. Shen, and F. Catthoor, "Implementation of fuzzy cognitive maps based on fuzzy neural network and application in prediction of time series," *IEEE Transactions on Fuzzy Systems*, vol.18, no.2, pp.233-250, 2010.
- [13] Z. Xu, Y. Liu, L. Mei, C. Hu, and L. Chen, "Generating temporal semantic context of concepts using web search engines," *Journal of Network & Computer Applications*, vol. 43, no. 1, 42-55,2014.
- [14] W. Stach, L. Kurgan, and W. Pedrycz, "Parallel learning of large fuzzy cognitive maps," in *Proc. Int. Joint Conf. Neural Networks*, pp.1584-1589, 2007.
- [15] X. Luo, X. Wei, and J. Zhang, "Guided game-based learning using fuzzy cognitive maps," *IEEE Transactions on Learning Technologies*, vol.3, no.4, pp.344-357, 2010.
- [16] N. Mateou, A. S. Andreou, and C. Stylianou, "A new traversing and execution algorithm for multilayered fuzzy cognitive maps," in *Proc. IEEE Int. Conf. Fuzzy Systems*, pp.2216-2223, 2008.
- [17] H. Zhuge, and Y. Xing, "Probabilistic resource space model for managing resources in Cyber-Physical society," *IEEE Transactions on Service Computing*, vol.5, no.3, pp.404-421, 2012.
- [18] J. Xuan, X. Luo, S. Zhang, et al. "Building hierarchical keyword level association link networks for web events semantic analysis," In *Proceedings of IEEE Ninth International Conference on Dependable Autonomic and Secure Computing*, pp. 987-994, IEEE, 2011.
- [19] X. Luo, Z. Xu, J. Yu, and X. Chen, "Building association link network for semantic link on web resources," *IEEE Transactions on Automation Science and Engineering*, vol.8, no.3, pp. 482-494, 2011.
- [20] X. Luo, J. Ni, J. Zhang, et al. "Building similar link network in large-scale web resources," In *Proceedings of IEEE 16th International Conference on Parallel and Distributed Systems*, pages 87-693, IEEE, 2010.
- [21] X. Wei, and X. Luo, "Concept extraction based on association linked network," In *Proceedings of International Conference on Semantics, Knowledge and Grids*, pp. 42-49, 2010.
- [22] X. Wei, X. Luo, Q. Li, and J. Zhang, "Online comment-based hotel quality automatic assessment using improved fuzzy comprehensive evaluation and fuzzy cognitive map," *IEEE Transactions on Fuzzy Systems*, vol.23, no.1, pp.72-84, 2015.
- [23] R. Agrawal, T. Imielinski, and A. Swami, "Mining association rules between sets of items in large databases," In *Proceedings of the ACM SIGMOD Conference on Management of Data*, pp.207-216, 1993.
- [24] Z. Xu, X. Wei, X. Luo, Y. Liu, L. Mei, and C. Hu, "Knowle: a semantic link network based system for organizing large scale online news events," *Future Generation Computer Systems*, vol. 43-44, pp.40-50, 2015.
- [25] Z. Xu, Y. Liu, N. Y. Yen, and L. Mei, "Crowdsourcing based description of urban emergency events using social media big data," *IEEE Transactions on Cloud Computing*, 10.1109/TCC.2016.2517638, in press.

Self-learning Change-prone Class Prediction

Meng Yan, Mengning Yang, Chao Liu, Xiaohong Zhang
School of Software Engineering
Chongqing University
Chongqing 401331, China
{meng.yan, mnyang, liu.chao, xhongz} @cqu.edu.cn

Abstract—Software change-prone class prediction can enhance software decision making activities during software maintenance (e.g., resource allocating). Many change-prone class prediction approaches have been proposed and most are effective in inter-version prediction within a project. These approaches usually build a supervised prediction model by learning from historical labeled dataset. However, a major challenge which remains is that this typical change-prone prediction setting cannot be used for new projects or projects with limited historical data. To address this challenge, we propose to tackle this task by adopting a novel prediction method which has not been used in change-prone prediction, namely self-learning method. The key idea of the self-learning method is to enable the change-prone prediction on new projects or projects with limited historical dataset by learning from itself. In this paper, we apply a state-of-art self-learning method, CLAMI, to change-prone prediction. In addition, we propose a novel self-learning approach CLAMI+ by extending CLAMI. The experiments among 14 open source projects show that the self-learning methods achieve comparable results to four typical inter-version baselines and the proposed CLAMI+ slightly improves the CLAMI method on average.

Keywords—software maintenance; change-prone prediction; self-learning; empirical software engineering

I. INTRODUCTION

Software maintenance has been regarded as one of the most expensive and tough tasks in the whole software lifecycle [1]. Managing and controlling changes in software maintenance is one of the significant concerns of the software industry [2]. A change could be made because of existence of bugs, new features or refactoring [3, 4]. A change-prone class means that the class is likely to change with a high probability after a product release. It can represent the weak part of a software system [2]. Thus, software change-prone class prediction contributes to better allocation of software resources (e.g., time and staff) in the software maintenance process [5]. This technique aids to support maintenance related decision making by identifying change-prone classes in advance. As a result, the quality assurance teams or testers can determine the critical parts of the software where the quality assurance or testing activities should pay more attention and track rigorously.

In order to predict change-prone classes in advance, various categories of software metrics have been proved to correspond to the change-proneness, such as OO metrics (e.g., cohesion, coupling, inheritance, etc.) [6], code smells [7], design patterns and [8] evolution metrics [9, 10]. In terms of the techniques,

different machine learning approaches have been used, such as Bayesian networks [11], neural networks [12], multivariate regression [1] and ensemble methods [5]. A typical prediction model based on machine learning is designed by learning from a historical labeled dataset in a supervised way. However, this technique is difficult to apply on new projects or projects with limited historical data.

A cross project change-prone class prediction method has been proposed to address the above-mentioned issue [13]. The cross project technique is motivated by the similar techniques in defect prediction [14, 15]. It enables change-prone class prediction on projects with limited labeled dataset by learning from other projects. Unfortunately, one issue which remains in cross-project prediction is that different datasets possess different distributions [16]. The success rate (ratio of combination whose performance is greater than a certain threshold) of cross-project reported in the work [13] is generally poor (30%) which cannot compare to the prediction performance (67%) of the methods using historical datasets (i.e., inter-version prediction within a project). This implies that the cross-project change-prone prediction may not be effective and it depends on the quality of the source project [13].

To address this issue, we propose to tackle this task by adopting self-learning method. In detail, we apply a state-of-art self-learning method (CLAMI: Clustering, LAbeling, Metric selection and Instance selection) to the change-prone class prediction which has been successfully used in defect prediction [16]. The key idea of this self-learning method is to enable the prediction on new projects or projects with limited labeled datasets by learning from itself.

The process of this self-learning method can be interpreted by dividing three phases as Figure 1 shows. The clue of the process is to build the prediction model by learning on selected informative metrics and instances of itself. In detail, the first phase is clustering and initialized labeling. In this phase, an unlabeled dataset is clustered and labeled according to the magnitude of metric values [16]. The motivation of this phase is to provide the initialized labels of all the instances. However, the initialized labels of all the instances might not be correct enough. In our self-learning method, some of them will be automatically selected as final training set according to our criteria in the following phase. The second phase is to conduct the metric selection and instance selection from the labeled instances in the first phase. As a result, an informative training set of metrics and instances are generated. The third phase is

modeling and prediction. The prediction model is built by learning from the selected instances in the second phase.

In particular, the initialized labeling step in the first phase of CLAMI method is conducted by measuring the count of violation (i.e., a metric value is greater than a certain threshold) of an instance. However, we observe that information loss might result from mapping the violation to a 1 or 0 (i.e., violation or not) result in the first phase. The information that how much the instance violated on a metric is not considered. Based on this observation, we propose a novel self-learning method CLAMI+ by extending CLAMI. The difference lies in the first phase as Figure 1 shows. In detail, the CLAMI+ method uses the violation degree (i.e., transforming the difference between the metric value and the threshold to a probabilistic value) to replace the Boolean representation in CLAMI. As a result, the fine information that how much the instance violated on a metric is considered. Under this way, the selection of final training set of CLAMI+ is different from CLAMI. The training set generated by CLAMI+ is expected more informative that CLAMI which is beneficial for building prediction model.

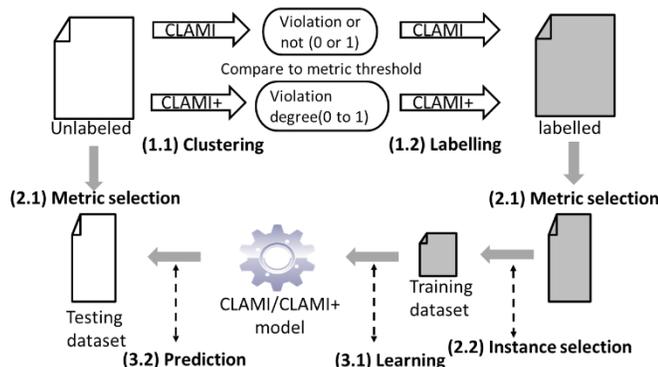


Figure 1. The overview of CLAMI/CLAMI+. It consists of three phases with two steps in each phase. The first phase is clustering and labeling (step 1.1 and 1.2), the CLAMI and CLAMI+ are different in this phase. The second phase is metric selection and instance selection (step 2.1 and 2.2). The third phase is learning and prediction (step 3.1 and 3.2).

The goal of our study is to conduct the change-prone class prediction in an automated way without the need of historical data. In our empirical experiments, we evaluate the self-learning methods on 14 open source projects which come from the Qualitas Corpus [17]. As a result, the self-learning methods yields a reasonable performance which improves the typical inter-version prediction models by 5.2%-19.7% (average CCR) and 13.9%-27.2% (average AUC), respectively. In addition, considering the average performance of all datasets, the proposed CLAMI+ method improves the CLAMI method by 2.9% (average CCR) and 3.2% (average AUC). In summary, the contributions of this study are as follows:

- We apply self-learning approach to tackle change-prone class prediction on new projects or projects with limited historical data. To the best of our knowledge, this is the first study to adopt self-learning approach in change-prone prediction. In addition, we propose a novel self-learning method CLAMI+ by extending the method CLAMI.

- We present an empirical study to evaluate the self-learning methods compared with typical inter-version change-prone class prediction methods on 14 public datasets.

II. RELATED WORK

A variety of approaches have been proposed to predict change-prone classes. For example, Amoui et al [12] proposed an innovative Neural Network-based temporal change prediction model which can predict where and when the change will happen. They achieved a reasonable performance on Mozilla and Eclipse. Godara et al [18] proposed an ID3 prediction model based on multi-factors. Koru et al. [19] first validated the Pareto’s law on change-prone classes on two open source projects, namely KOffice and Mozilla. They found the applicability of Pareto’s law and developed a tree based prediction model. Lu et al [20] proposed a statistical meta-analysis approach to explore the ability of 62 OO metrics for predicting change-proneness on 102 Java systems. They found that size metrics were more discriminative than other OO metrics, such as cohesion, coupling and inheritance. Elish et al [5] proposed an empirical study which used ensemble methods on change prediction. They found that ensemble methods can achieve a better performance than individual models. However, one issue in the above-mentioned works is that the prediction model relies on learning from the historical data in a supervised way, such as learning from the labeled data from previous project version or learning from labeled data within a project version (i.e., cross validation). It is difficult to apply the technique on new projects or projects with limited historical data.

Cross project prediction is a solution to address the above-mentioned limitation. The cross project concept is introduced by Briand et al [21]. It has been widely used in defect prediction [14, 22-25]. In change-prone class prediction, there are also a few studies which have investigated the cross-project change prediction recently.

Malhotra et al [13] proposed to build the cross project change prediction model by using the logitboost method. In another work, Malhotra et al [26] validated the cross project change prediction by using machine learning and search-based techniques. However, they found that the cross project (or inter project) prediction cannot comparable to the inter-version prediction within a project (i.e., learning from previous version and testing on the current version) [13]. Besides, one main issue remains in cross-project prediction is that different projects possess different data distributions [16]. How to select an appropriate source as the training data is a difficult task [13].

To address the above-mentioned limitation, the self-learning method can enable the prediction task which does not need a prior labeled source as the training dataset. In other words, the self-learning method leads to building the change prediction model through learning by itself.

III. APPROACH

This section describes the process of the self-learning approach. It consists of three phases and we describe the three phases in three subsections (subsection A, B and C). The first

phase is Clustering and Labeling, the second phase is Metric selection and Instance selection, the third phase is Learning and Prediction. In particular, the idea of the first phase in CLAMI+ is different with CLAMI, and the idea of the second phase and the third phase in CLAMI+ is identical with CLAMI.

A. Clustering and Labeling

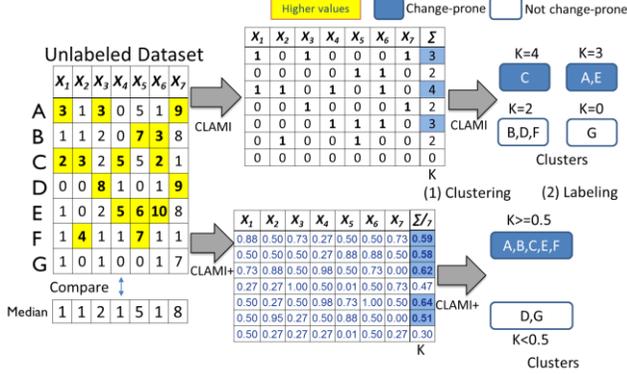


Figure 2. The process of the first phase in the self-learning approach. Higher values mean the metric value is greater than the median.

1) *Clustering*: The key idea of the clustering process in the self-learning approach is shown in Figure 2. We use A-G to denote the instances of the dataset and x_1 - x_7 denote the adopted metrics. A specific cutoff threshold is set as the median value for each metric as Nam et al [16] described. The first step is to compare the metric value to the threshold value for each metric. As a result, a violation table is generated.

In CLAMI, the violation table consists of 0 or 1 values. The value “1” represents a higher value which means it is greater than the threshold value as highlighted in Figure 2. After that, the clustering process groups the instances by the sum (K value) of the count of the higher values. For example, the instances A and E belong to one cluster ($K=3$) which means there are three higher values in A and E. However, one issue remains is that the information that how much the instance violated on a metric is not considered. For example, considering instance B and E at the metric x_6 , the metric value of B is 3 and the metric value of E is 10. Although both of them are violated values which is greater than threshold 1, the violation degree of instance E is greater than instance B obviously. This information is ignored in CLAMI.

In CLAMI+, we extend the CLAMI approach by transforming the 1 or 0 result (violation or not) to a continuous value from 0 to 1 which represents the violation degree. As a result, the violation table consists of continuous values ranging from 0 to 1 as Figure 2 shows. In detail, we adopt the sigmoid function which is often used as the activation function in neural networks to conduct the probabilistic transformation. Formally, suppose there are M instances and N metrics, X_{ij} denotes the j -th metric value of the i -th instance, N_j denotes the threshold value of the j -th metric. The violation degree of the j -th metric of the i -th instance $P(V_{ij})$ is computed as Formula (1). Different from the CLAMI, the κ value in CLAMI+ represents

the mean violation degree of an instance and we group the instances by $K \geq 0.5$ and $K < 0.5$.

$$P(V_{ij}) = \frac{1}{1 + e^{-(X_{ij} - N_j)}} \quad (1)$$

2) *Labeling*: In CLAMI, the labeling step is conducted by dividing the clusters into a top half and a bottom half by considering the K value [16]. Next, the first half clusters are labeled as change-prone and the bottom half clusters are labeled as not change-prone. Similar to CLAMI, in CLAMI+, we label the instances by dividing the clusters into $K \geq 0.5$ and $K < 0.5$. Next, we label the first cluster as change-prone and the second cluster as not change-prone. As the Figure 2 shows, in CLAMI, Instance C, A and E are labeled as change-prone while in CLAMI+ Instance A, B, C, E and F are labeled as change-prone. This difference is resulted from our usage of the violation degree.

The labeling step is motivated by the tendency in defect prediction, namely, the defect-prone instances have higher metric values than clean-prone instances [16, 27]. Since the typical metrics which are adopted in both defect prediction and change prediction (e.g., OO metrics and general size metrics) represent the complexity of the instance, there is also the similar tendency in change-prone prediction [28, 29] (named as change-prone tendency). For example, Koru et al [28] found that that high-change modules had fairly high places in metric rankings, although not the highest places. Malhotra et al. [29] found that the classes whose metric values exceed a threshold value are change prone. Therefore, we label the top half or the first cluster instances as change-prone.

B. Metric Selection and Instance Selection

In order to generate a high-quality training set, we use metric and instance selection to select informative metrics and minimize the instances that may be incorrectly labeled in the first phase.

1) *Metric Selection*: The quality of features plays a significant role in building a prediction model. Since there might be some metrics which do not follow the change-prone tendency well, the objective of metric selection step is to select the most informative metrics which can enhance the prediction ability. The selection criteria is the metric violation scores (MVS) for each metric. In terms of one metric, the MVS is equal to the count of instances which do not follow the change-prone tendency on this metric. Take the metric x_1 in Figure 2 as the example, instance B is labeled as a change-prone instance in the first phase of CLAMI+, however, the metric value of x_1 is not a higher value, thereby B does not follow the tendency at metric x_1 . Using this way, we compute the MVS for each metric and select the metrics which have the minimum MVS.

2) *Instance Selection*: In order to generate a better training set, instance selection is a widely adopted technique in software prediction models [30, 31]. It is the final step for generating the training dataset in this self-learning method. In detail, we select the instances which follow the change-prone tendency at the selected metrics. In other words, we remove

the instances which do not follow the change-prone tendency on the selected metrics. For example, suppose x_1 is a selected metric in Figure 2 and B is labeled as a change-prone instance in the first phase of CLAMI+. However, the metric value of x_1 does not follow the change-prone tendency (the metric value is expected to greater than threshold) in Instance B, thereby we will remove B from the final training set. After this step, in some cases which have too many tendency-violated instances, there might be no change-prone or not change-prone instances. In this sense, we will get back to the metric selection step and choose extra metrics which have the next minimum MVS until both change-prone and not change-prone instances exist in the training set.

C. Learning and Prediction

After generating a training set, we adopt a general machine learner (logistic regression) to build the prediction model which learns from the selected metrics and instances. By the following, we predict the change-prone classes of the testing set on the selected metrics.

IV. EXPERIMENTAL DESIGN

A. Research Questions

We design two research questions to evaluate this study. One is to evaluate the performance of self-learning method. The other is to evaluate the effectiveness of this proposed novel self-learning method CLAMI+.

- **RQ1:** *Is the prediction performance of the self-learning methods comparable to typical prediction methods based on historical data?* The advantage of the self-learning method over typical prediction methods is that it does not need historical labeled dataset. We will answer this question by comparing the self-learning method and the typical prediction methods on the same target dataset. The difference is that the typical prediction methods learn from historical labeled dataset while our self-learning method learn from itself.
- **RQ2:** *Does the prediction performance of CLAMI+ outperform CLAMI?* The difference between CLAMI and CLAMI+ is the criteria of clustering and labeling. As a result, the training set is different which has an impact on the prediction performance. We will answer this question by comparing the two methods on the same datasets used in RQ1.

B. Datasets

We evaluate this study on 14 open source projects which come from the public dataset Qualitas Corpus [17] (Qualitas Corpus version is 20130901e). They are written in Java and have multiple evolution versions. For each project, we choose the recent version as the target dataset and label each instance by tracking the version control system. The target project version, previous version (used in baselines), percentage of changed instances and the total number of instances in the target version are listed in Table I. The percentage and the

number of instances possess a substantial range which can validate the model ability among a wide range.

TABLE I: SUMMARY OF THE EVALUATION DATASETS IN THIS STUDY

Previous version	Target version	% changed	# instances
'ant-1.8.1.0'	'ant-1.8.2.0'	12.20%	844
'antlr-3.3.0'	'antlr-3.4.0'	70.95%	241
'argouml-0.32.1'	'argouml-0.32.2'	39.67%	1505
'azureus-4.1.0.2'	'azureus-4.1.0.4'	7.71%	3150
'freecol-0.10.4'	'freecol-0.10.5'	71.74%	598
'freemind-0.6.5'	'freemind-0.6.7'	89.19%	74
'hibernate-3.1.1.0'	'hibernate-3.1.2.0'	93.62%	925
'jgraph-5.12.0.4'	'jgraph-5.12.1.0'	20.75%	53
'jmeter-2.7.0.0'	'jmeter-2.8.0.0'	58.07%	830
'jstock-1.0.7.1'	'jstock-1.0.7.2'	11.59%	276
'jung-1.7.2'	'jung-1.7.4'	28.85%	468
'junit-4.9.0'	'junit-4.10.0'	92.02%	163
'lucene-3.6.2.0'	'lucene-4.0.0.0'	34.35%	620
'weka-3.5.7'	'weka-3.5.8'	13.94%	1119

Considering the code metrics, we adopt the typical metrics which are identical with the relevant studies of change prediction [1, 5, 11, 32] as the Table II shows. In detail, five Chidambar and Kemerer metrics [33]: WMC, DIT, NOC, RFC, and LCOM; four Li and Henry metrics [34]: MPC, DAC, NOM, SIZE2; and one traditional lines of code metric (SIZE1) are adopted. SIZE1 represents the number of lines of code excluding comments and SIZE2 represents the total count of the number of data attributes and the number of local methods in a class.

TABLE II: SUMMARY OF THE ADOPTED METRICS IN THIS STUDY

Metric	Description
WMC	Count of methods implemented within a class
DIT	Level for a class within its class hierarchy
NOC	Number of immediate subclasses of a class
RFC	Count of methods implemented within a class plus the number of methods accessible to an object class due to inheritance
LCOM	The average percentage of methods in a class using each data field in the class subtracted from 100 %
MPC	The number of messages sent out from a class
DAC	The number of instances of another class declared within a class
NOM	The number of methods in a class
SIZE1	The number of lines of code excluding comments
SIZE2	The total count of the number of data attributes and the number of local methods in a class

C. Experimental Baselines

In RQ1, we set the typical inter-version prediction methods [13] within a project as baselines to compare with the self-learning methods. In other words, in order to predict the change-prone classes of the target version, the prediction models of the baselines are built by learning from the labeled dataset of the previous release version. In our experiment, the target version is as Table I shows and we set the previous neighbor version of the target version as the training source in the baselines. The first baseline is the change-prone class prediction based on logitboost (LB) which is proposed by Malhotra et al [13]. In addition, to avoid the bias from only one method, we also adopt three typical machine learners as baselines which are used in all three empirical studies on

TABLE III: PERFORMANCE COMPARISON BETWEEN SELF-LEARNING METHODS AND FOUR INTER-VERSION PREDICTION BASELINES. IF THE PERFORMANCE OF THE SELF-LEARNING METHODS CLAMI/CLAMI+ OUTPERFORMS ALL THE FOUR BASELINES, THE RESULTS ARE IN BOLD. THE BETTER RESULTS BETWEEN CLAMI AND CLAMI+ ARE UNDERLINED.

Project	CCR						AUC					
	LB	MLP	RBF	SVM	CLAMI	CLAMI+	LB	MLP	RBF	SVM	CLAMI	CLAMI+
'ant-1.8.2.0'	88.63	89.22	88.63	87.80	58.29	58.29	0.60	0.60	0.58	0.51	0.65	0.65
'antlr-3.4.0'	56.85	53.53	46.47	36.51	65.56	65.56	0.63	0.60	0.55	0.47	0.65	0.65
'argouml-0.32.2'	60.33	60.33	60.33	60.33	46.05	<u>52.23</u>	0.51	0.51	0.51	0.51	0.49	<u>0.52</u>
'azureus-4.1.0.4'	92.32	92.38	92.29	92.29	52.83	<u>52.83</u>	0.47	0.47	0.47	0.47	0.64	0.64
'freecol-0.10.5'	29.93	30.10	28.26	28.26	63.88	64.05	0.47	0.47	0.45	0.45	0.67	0.67
'freemind-0.6.7'	39.19	43.24	32.43	24.32	52.70	60.81	0.66	0.62	0.56	0.58	0.61	<u>0.63</u>
'hibernate-3.1.2.0'	6.92	8.22	6.38	6.38	50.70	57.62	0.54	0.55	0.54	0.54	0.61	0.62
'jgraph-5.12.1.0'	84.91	86.79	88.68	83.02	69.81	<u>69.81</u>	0.65	0.67	0.73	0.49	0.72	0.77
'jmeter-2.8.0.0'	52.89	57.59	51.08	50.84	63.98	66.63	0.56	0.60	0.56	0.58	0.65	0.67
'jstock-1.0.7.2'	89.49	89.49	89.13	88.41	55.07	<u>55.07</u>	0.61	0.61	0.60	0.57	0.62	0.66
'jung-1.7.4'	72.65	72.01	71.58	71.15	<u>52.35</u>	51.50	0.49	0.48	0.47	0.47	0.51	0.56
'junit-4.10.0'	12.88	14.72	9.20	7.98	52.76	52.76	0.57	0.59	0.55	0.55	0.76	0.76
'lucene-4.0.0.0'	34.35	34.35	34.35	34.35	65.97	65.97	0.47	0.47	0.47	0.47	0.65	0.65
'weka-3.5.8'	36.10	33.87	37.62	21.00	55.59	55.67	0.52	0.48	0.53	0.45	0.59	0.58
<i>Average</i>	54.10	54.70	52.60	49.47	57.54	59.20	0.55	0.55	0.54	0.51	0.63	0.65

change prediction proposed by Elish et al [5]. Therefore, the second, third and the fourth baseline is Multilayer perceptron (MLP), Radial basis function network (RBF) and Support vector machine (SVM), respectively.

In RQ2, we compare the extended CLAMI+ to the original CLAMI method proposed by Nam et al [16].

D. Performance Measures

Same as in the work of Elish et al [5], two widely used prediction measures are adopted in our evaluation, namely correct classification rate (CCR) and the area under curve (AUC). CCR represents the ratio of cases which were correctly predicted to the total number of cases. AUC represents the area under the receiver operating characteristic (ROC) curve.

V. RESULTS

Table III shows the performance comparison between the self-learning methods and the baselines in CCR and AUC under 14 datasets. In terms of each dataset, if the performance of the self-learning methods CLAMI/CLAMI+ outperforms all four baselines, the results are bold. The better results between CLAMI and CLAMI+ are underlined.

Overall, in terms of RQ1, the self-learning methods CLAMI and CLAMI+ show comparable performance to the four inter-version prediction methods. In particular, considering the CCR measure, the self-learning methods outperform the four baselines among 8 datasets. In the dataset like ant-1.8.2.0, the self-learning methods perform worse. However, considering the average of all datasets, self-learning methods CLAMI/CLAMI+ improve them by 5.2%-19.7%. Considering the AUC measure, self-learning methods CLAMI/CLAMI+ outperform four baselines among 11 datasets and improve them by 13.9%-27.2% in average of all datasets. Note that the self-learning methods do not need prior labeled data but achieve comparable performance than inter-version prediction methods. In terms of RQ2, the performance of CLAMI+ achieves comparable or better result than CLAMI method. Only one out of the 14 datasets in which the CLAMI+ shows

the worse result than CLAMI considering AUC or CCR. In other cases, the CLAMI+ performs better or at least the same with CLAMI. Also, considering the average of all datasets, the CLAMI+ method improves the CLAMI method by 2.9% in CCR and 3.2% in AUC.

In addition, we conduct the Friedman test on the performance comparison when comparing multiple methods as suggested by Demša [35]. The Friedman test compares whether the difference of the average ranks of the performance of the methods are statistically significant or not. We translate the question into the null hypothesis H_{null} : There is no significant difference between the average ranks of the performance of all the methods. And the alternative hypothesis H_{alt} is that there is a significant difference between the average ranks of all the methods. Table IV shows the Friedman test results. We provided the average ranks (the approach with the best performance is ranked in "6") and the significant level p -value. In terms of CCR, the CLAMI and CLAMI+ show a comparable ranks although not the best. In terms of AUC, the CLAMI and CLAMI+ show the higher ranks than other four methods and the CLAMI+ is the best. Note that the p -values in both of the two measures are less than 0.05 which enable us to reject the null hypothesis and accept the alternative hypothesis. This indicates that there is a statistical significant difference between the average ranks of all the methods in the two performance measures.

TABLE IV: FRIEDMAN TEST FOR THE PERFORMANCE COMPARISON

Measure	Average rank						p -value
	LB	MLP	RBF	SVM	CLAMI	CLAMI+	
CCR	3.86	4.25	3.14	2.18	3.61	3.96	0.0356
AUC	3.18	3.29	2.32	1.79	4.79	5.64	0.0000

VI. THREATS TO VALIDITY

Impact of the threshold. Threshold decides the results of the clustering and initialized labeling phase. There are various methods to decide a metric threshold. However, we did not provide the analysis on the impact of different thresholds. This might be a threat to our work. In this work, we adopt a typical threshold (i.e., the median) to mitigate this issue. A more

refined work is to take into account the effects of different thresholds.

Impact of sigmoid function. The difference of our proposed CLAMI+ method is that we transform the 1 or 0 result (violation or not) to a continuous value ranging from 0 to 1 which represents the violation degree by using the sigmoid function. There are several parameters in a sigmoid function, such as dynamic range, and slope. However, we adopt the regular sigmoid function on all the metrics. This might be a limitation to the performance of CLAMI+, since different metrics possess different distribution and they may be suitable for different parameter settings. Also, we have a plan to conduct additional experiment on the impact of the parameters.

VII. CONCLUSION AND FUTURE WORK

In this paper, we proposed to adopt self-learning approach to tackle change-prone class prediction on new projects or projects with limited historical data. Concretely, we applied a state-of-art self-learning method CLAMI and proposed a novel approach CLAMI+ by extending CLAMI on change-prone class prediction. This enables prediction for new projects or projects with limited historical data. The empirical study among 14 open source projects showed that the self-learning methods yield better or comparable results to four typical inter-version prediction methods in terms of CCR and AUC. In addition, the proposed CLAMI+ method slightly improves the CLAMI method on average.

In the future, we plan to enhance the effectiveness of our approach further. Concretely, we plan to investigate the impact of various thresholds, such as mean, standard deviation and different percentiles. In addition, we plan to improve the performance by proposing an adaptive method to determine the optimized parameters of the sigmoid function for different metrics.

ACKNOWLEDGMENTS

The work described in this paper was partially supported by the National Natural Science Foundation of China (Grant no. 91118005, 61173131, 11202249), Chongqing Graduate Student Research Innovation Project (grant no. CYS14008 and CYS15022), Program for Changjiang Scholars and Innovative Research Team in University (Grant No. IRT1196) and the Fundamental Research Funds for the Central Universities of China (Grant No. 106112014CDJZR098801).

REFERENCES

- [1] Y. Zhou and H. Leung, "Predicting object-oriented software maintainability using multivariate adaptive regression splines," *Journal of Systems and Software*, vol. 80, pp. 1349-1361, 2007.
- [2] R. Malhotra and M. Khanna, "Examining the effectiveness of machine learning algorithms for prediction of change prone classes," in *Proceedings of the International Conference on High Performance Computing & Simulation*, 2014, pp. 635-642.
- [3] M. Yan, Y. Fu, X. Zhang, D. Yang, L. Xu, and J. D. Kymer, "Automatically classifying software changes via discriminative topic model: Supporting multi-category and cross-project," *Journal of Systems and Software*, vol. 113, pp. 296-308, 2016.
- [4] Y. Fu, M. Yan, X. Zhang, L. Xu, D. Yang, and J. D. Kymer, "Automated classification of software change messages by semi-supervised Latent Dirichlet Allocation," *Information and Software Technology*, vol. 57, pp. 369-377, 2015.
- [5] M. Elish, H. Aljamaan, and I. Ahmad, "Three empirical studies on predicting software maintainability using ensemble methods," *Soft Computing*, vol. 19, pp. 2511-2524, 2015/09/01 2015.
- [6] Z. Yuming, H. Leung, and X. Baowen, "Examining the Potentially Confounding Effect of Class Size on the Associations between Object-Oriented Metrics and Change-Proneness," *IEEE Transactions on Software Engineering* vol. 35, pp. 607-623, 2009.
- [7] F. Khomh, M. Di Penta, Gue, x, he, x, et al., "An Exploratory Study of the Impact of Code Smells on Software Change-proneness," in *Proceedings of the 16th Working Conference on Reverse Engineering (WCRE 2009) 2009*, pp. 75-84.
- [8] D. Posnett, C. Bird, and P. Dévanbu, "An empirical study on the influence of pattern roles on change-proneness," *Empirical Software Engineering*, vol. 16, pp. 396-423, 2011/06/01 2011.
- [9] M. O. Elish and M. Al-Rahman Al-Khiaty, "A suite of metrics for quantifying historical changes to predict future change-prone classes in object-oriented software," *Journal of Software: Evolution and Process*, vol. 25, pp. 407-437, 2013.
- [10] S. Eski and F. Buzluca, "An Empirical Study on Object-Oriented Metrics and Software Evolution in Order to Reduce Testing Costs by Predicting Change-Prone Classes," in *Proceedings of the IEEE Fourth International Conference on Software Testing, Verification and Validation Workshops*, 2011, pp. 566-571.
- [11] C. van Koten and A. R. Gray, "An application of Bayesian network for predicting object-oriented software maintainability," *Information and Software Technology*, vol. 48, pp. 59-67, 2006.
- [12] M. Amoui, M. Salehie, and L. Tahvildari, "Temporal software change prediction using neural networks," *International Journal of Software Engineering and Knowledge Engineering*, vol. 19, pp. 995-1014, 2009.
- [13] R. Malhotra and A. J. Bansal, "Cross project change prediction using open source projects," in *Proceedings of the International Conference on Advances in Computing, Communications and Informatics*, 2014, pp. 201-207.
- [14] J. Nam, S. J. Pan, and S. Kim, "Transfer defect learning," presented at the *Proceedings of the International Conference on Software Engineering*, San Francisco, CA, USA, 2013.
- [15] A. Panichella, R. Oliveto, and A. De Lucia, "Cross-project defect prediction models: L'Union fait la force," in *Proceedings of the IEEE Conference on Software Maintenance, Reengineering and Reverse Engineering*, 2014, pp. 164-173.
- [16] J. Nam and S. Kim, "CLAMI: Defect Prediction on Unlabeled Datasets," in *Proceedings of the 30th IEEE/ACM International Conference on Automated Software Engineering*, 2015, pp. 452-463.
- [17] E. Tempero, C. Anslow, J. Dietrich, T. Han, L. Jing, M. Lumpe, et al., "The Qualitas Corpus: A Curated Collection of Java Code for Empirical Studies," in *Proceedings of the 17th Asia Pacific Software Engineering Conference*, 2010, pp. 336-345.
- [18] D. Godara and R. Singh, "A New Hybrid Model for Predicting Change Prone Class in Object Oriented Software," *International Journal of Computer Science and Telecommunications*, vol. 5, pp. 1-6, 2014.
- [19] A. Güneş Koru and H. Liu, "Identifying and characterizing change-prone classes in two large-scale open-source products," *Journal of Systems and Software*, vol. 80, pp. 63-73, 2007.
- [20] H. Lu, Y. Zhou, B. Xu, H. Leung, and L. Chen, "The ability of object-oriented metrics to predict change-proneness: a meta-analysis," *Empirical Software Engineering*, vol. 17, pp. 200-242, 2012/06/01 2012.
- [21] L. C. Briand, W. L. Melo, and J. Wust, "Assessing the applicability of fault-proneness models across object-oriented software projects," *IEEE Transactions on Software Engineering*, vol. 28, pp. 706-720, 2002.
- [22] Z. He, F. Shu, Y. Yang, M. Li, and Q. Wang, "An investigation on the feasibility of cross-project defect prediction," *Automated Software Engineering*, vol. 19, pp. 167-199, 2012/06/01 2012.

- [23] F. Peters, T. Menzies, and A. Marcus, "Better cross company defect prediction," in Proceedings of the 10th IEEE Working Conference on Mining Software Repositories, 2013, pp. 409-418.
- [24] T. Zimmermann, N. Nagappan, H. Gall, E. Giger, and B. Murphy, "Cross-project defect prediction: a large scale experiment on data vs. domain vs. process," presented at the Proceedings of the 7th joint meeting of the European software engineering conference and the ACM SIGSOFT symposium on The foundations of software engineering, Amsterdam, The Netherlands, 2009.
- [25] B. Turhan, T. Menzies, A. B. Bener, and J. Di Stefano, "On the relative value of cross-company and within-company data for defect prediction" Empirical Software Engineering, vol. 14, pp. 540-578, 2009.
- [26] R. Malhotra and M. Khanna, "Mining the impact of object oriented metrics for change prediction using Machine Learning and Search-based techniques," in Proceedings of the International Conference on Advances in Computing, Communications and Informatics, 2015, pp. 228-234.
- [27] T. Menzies, J. Greenwald, and A. Frank, "Data Mining Static Code Attributes to Learn Defect Predictors," IEEE Transactions on Software Engineering, vol. 33, pp. 2-13, 2007.
- [28] A. G. Koru and J. Tian, "Comparing high-change modules and modules with the highest measurement values in two large-scale open-source products," IEEE Transactions on Software Engineering vol. 31, pp. 625-642, 2005.
- [29] R. Malhotra and A. Bansal, "Prediction of Change Prone Classes using Threshold Methodology," Advances in Computer Science and Information Technology, vol. 2, pp. 30-35, 2015.
- [30] E. Kocaguneli, T. Menzies, J. Keung, D. Cok, and R. Madachy, "Active learning and effort estimation: Finding the essential content of software effort estimation data," IEEE Transactions on Software Engineering, vol. 39, pp. 1040-1053, 2013.
- [31] Y. F. Li, M. Xie, and T. N. Goh, "A study of project selection and feature weighting for analogy based software cost estimation," Journal of Systems and Software, vol. 82, pp. 241-252, 2009.
- [32] M. O. Elish and K. O. Elish, "Application of TreeNet in Predicting Object-Oriented Software Maintainability: A Comparative Study," in Proceedings of the 13th European Conference on Software Maintenance and Reengineering (CSMR 2009), 2009, pp. 69-78.
- [33] S. R. Chidamber and C. F. Kemerer, "A metrics suite for object oriented design," IEEE Transactions on Software Engineering, vol. 20, pp. 476-493, 1994.
- [34] W. Li and S. Henry, "Object-Oriented metrics that predict maintainability," Journal of Systems and Software, vol. 23, pp. 111-122, 1993.
- [35] J. Demsar, "Statistical Comparisons of Classifiers over Multiple Data Sets," J. Mach. Learn. Res., vol. 7, pp. 1-30, 2006.

Generating Summarized Preview for Education Resource based on Exploring and Comparing GUIs

Chao Ma

School of Data and
Computer Science,
Sun Yat-sen University,
Guangzhou, China
Research Institute of Sun
Yat-sen University in
Shenzhen, Shenzhen, China
qingyuanluofeng@163.com

Xiangping Chen*

Institute of Advanced
Technology,
Sun Yat-sen University,
Guangzhou, China
Research Institute of Sun
Yat-sen University in
Shenzhen, Shenzhen, China
chenxp8@mail.sysu.edu.cn

Yongsheng Rao

School of Data and
Computer Science,
Sun Yat-sen University,
School of Computer Science
and Educational Software,
Guangzhou University,
Guangzhou, China
rysheng@163.com

Mouguang Lin

Institute of Advanced
Technology,
National Engineering
Research Center of Digital
Life,
Sun Yat-sen University,
Guangzhou, China
garnettlam@126.com

Abstract—Compared to traditional education resource, Interactive Education Resource (IER) includes dynamic content to interact with users in order to enhance comprehension. The contents of IERs are relatively complex because of its dynamic feature. Exploring IERs and understanding its content during resource search become time-consuming. In this paper, we propose a preview generation approach based on automated software testing and image processing. Our approach first analyzes IERs to collect interactive elements and the event list. Based on the interaction related information, our approach simulates user actions and record snapshots step by step. Our approach compares snapshots to generate an image including the basic content, interaction region and dynamic contents. The image is used as the overall preview and the animation of snapshots collected during simulation is used as detail preview of the IER. In the experiment, we evaluate the completeness, correctness, and comprehension of the preview. The experiment result shows that the preview can be used to provide most information in the resource correctly and easy to understand. Our approach can save time for users in understanding the IERs.

Keywords- *preview generation; interactive education resource; dynamic content collage*

I. INTRODUCTION

E-learning entails the use of electronic technologies in learning and teaching. With the increasing popularity of e-learning, techniques and tools for developing interactive education resources are developed to enhance inspiration, such as Beam[1] and 3D tangible model[2]. Because this kind of education resources can enhance user interaction to improve understanding of knowledge, a wide variety of interactive educational resources are developed.

With the increasing number of interactive educational resources, searching resources become time consuming. Firstly, an interactive educational resource usually depends on a specific tool. Some of the tools for browsing interactive educational resources do not have an online version. Users are required to download the resource and its browsing tool, install the tool to open the resource, explore the resource to find out whether the resource is valuable. Secondly, the cost to understand the interactive content is relatively high. Text

information in most interactive resources is easy to understand because it usually does not involve long text. However, locating interactive elements and triggering user action are time-consuming for most readers. For the users who are unfamiliar with the way to interact with this kind of education resource, the learning cost increases.

Current research on resource preview provides approaches to save time in understanding resource with complex content, such as web page[3,4,7] and interaction[5,6]. It can avoid downloading and installing unnecessary tool and resources and save time in understanding content. Image[7], text[8], video[9,10], audio[11,12] are used to provide summarized preview and proven to be useful in practice.

However, there is no existing preview generation approaches for interactive educational resource. Considering the importance of interaction, understanding a resource includes understanding how to interact with this resource and knowing what dynamic content will appear during interaction.

In this paper, we propose an approach for generating summarized preview for interactive education resource. The summarized preview is an image which collages the basic content, interaction region and dynamic contents of the resource. In order to collect content of the resource, an interactive education resource opened by its exploring tool is viewed as a knowledge specific application. Our approach generates event list by analyzing interactive elements in the resource. Based on the event set, our approach traverses the resource and collect snapshots by triggering operations related to interactive elements. Our approach uses Heatmap algorithm to render interaction region. And then, dynamic contents are extracted by comparing snapshots and clustering changes in different snapshots. The first snapshot rendered in the interaction region and image collaged with dynamic content are used as the overall preview. The animation of snapshots collected during simulation is used as detail preview of the IER.

In case study, we randomly select 20 interactive education resource of the type Super Sketchpad[21] to valid our approach. 15 Students from Sun Yat-sen University are invited to evaluate preview effect according to its completeness,

correctness, and comprehension. The average scores of the preview completeness, correctness and comprehension ratings are 4.42, 4.53 and 4.28 (max is 5). The experiment result shows that the preview is valuable for users. The summarized preview can save 54.89% time for users on average. Users can understand 69.04% of content using the preview provided by our approach.

The remainder of the paper is organized as follows: Section 2 gives a general overview of our solution, which is further detailed in the Section 3. Section 4 presents the evaluation of our approach; Section 5 discusses some related works before Section 6 concludes our work.

II. APPROACH OVERVIEW

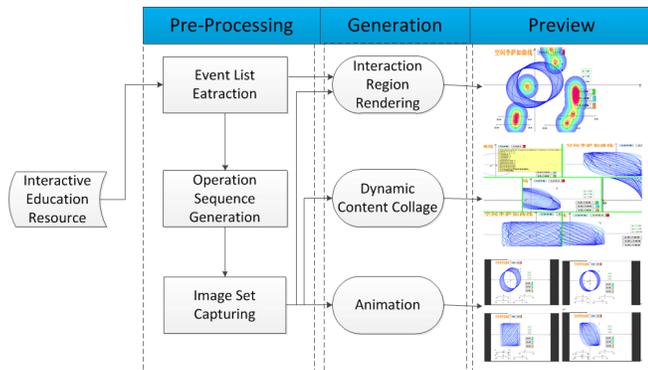


Figure 1. Approach Overview

Figure 1 shows the overview of our approach for generating summarized preview of IERs. In order to collect content of the resource, an interactive education resource opened by its exploring tool is viewed as a knowledge specific application. Inspired by GUI centric testing techniques, our approach collects event list related to the interactive element. And then, operations are extracted according to event types and operated in the GUI. Snapshots are recorded during the simulation.

The summarized preview is an image which collages the basic content, interaction region and dynamic contents of the resource. We take the first snapshot collected in the pre-processing stage as the basic content of the resource. In rendering interaction region, Heatmap algorithm is used with Gaussian model for color weight computation. And then, dynamic contents are extracted by comparing snapshots and clustering changes in different snapshots. The first snapshot rendered in the interaction region and image collaged with dynamic content are used as the overall preview. Considering that some details of the resource are missing in the summarized preview, the animation of snapshots collected during simulation is used as supplement.

III. GENERATING SUMMARIZED PREVIEW

A. Pre-processing

Interactive education resource are developed to facilitate understanding of knowledge during learning activities. The interaction between these resource and users are relatively simpler compared with the interaction types supported in applications.

- Education resource may include one or more pages. When there are more than one page, pages are usually arranged in sequence.
- The interaction types are decided by the resource definition language and exploration tool. The number of event types are usually less than the number of event types supported by programming languages.

Based on these two assumptions, we model the interaction between interactive resource and users. An interactive resource is $IR=(P, R)$, P is a set of pages, R is the sequence relationship between pages. A page can be abstracted as $P=(E, DR)$. E is a set of elements in the page, and DR is the dependency relationship between two elements. $DR=\{ \langle e_i, e_j \rangle | e_i \in E \wedge e_j \in E \}$. An element is abstracted as $e=\langle id, eType, data, actionType, scope, reactionNum \rangle$. id is the identification of an element in a page. $data$ is the data associated with the element. $eType$ is used to define the interactive and visible feature of elements. There are three types: invisible, visible&interactive, visible&static. $actionType$ is the type of action to input through computer peripherals, such as keyboard input, mouse click etc. $scope$ of an action defines the positions in the interface in which the action can be executed. $reactionNum$ is the number of reaction according to user action.

In the pre-processing stage, the information for describing an interactive resource is extracted based on the API provided by resource exploring tools. In this stage, the implementation is highly related to kind of resource. Two kinds of information are required: the events related to interactive elements and the relationships between elements. For the former information, the extraction is relatively simpler because information of interactive elements are usually directly recorded in the resource file. The implementation of extraction can be done by analyzing source file according to the file format.

The extracting relationships between interactive elements are more complex because (1) relationships are usually not recorded explicitly. (2) the classification of relation types may differ in different resource exploring tools. The mismatch requires implementing adapters in practice. It is worth noticing that our approach can generate preview even when there is a lack of relationship information. Dependency relationships are used to enhance the quality of operation sequence.

The operation sequence S is a set of *event* which are triggered in the user interface. *event* is defined as $event = \langle eid, actionType, position \rangle$. An *event* is an execution of action on an interactive element. We generate operation sequence S which triggers events of all the interactive elements at least once and meet the constraints of element dependency. Based on the operation sequence, an execution script can be generated. In our implementation, we generate script AutoHotkey, which is a free, open-source macro-creation and automation software for Windows that allows users to automate repetitive tasks[13]. Based on the execution script, AutoHotkey executes the operation sequence according the *actionType*, *position* and *reactionNum*. When an operation is executed, the snapshot is recorded and saved.

Considering animation, storyboard can be used to show teaching example visually which can strengthen the comprehension of computing theory[14], we choose animation as one of the preview methods for IER. Animation can show the change of IER when each operation is executed. We write an automated playing program to play the image set of snapshots setting internal time as 1 second. Figure 2 shows the animation preview of an IER which is made by the tool Super Sketchpad and used to introduce the math knowledge about Lissajous-Figur. This example is also used in the following section.

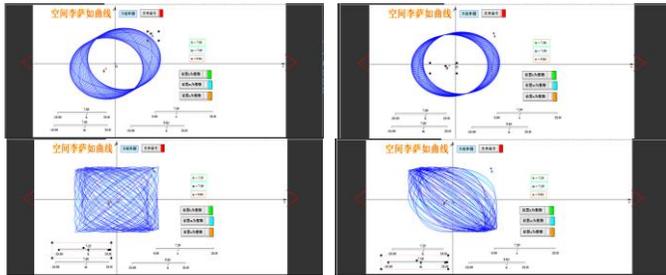


Figure 2. Animation Preview During 4 Seconds

B. Preview Generation

The generation of preview includes rendering interaction region in the basic content and collaging dynamic contents.

1) Interaction Region Rendering

Rendering interaction region is used to indicate users how to interact with the resource and avoid false interaction. Firstly, whether an element is interactive or not is not decided by its visual appearance. Interaction region can be used to distinguish interactive elements. Secondly, small size interactive elements will not be missed with explicitly rendering.

Heatmap[15] can render objects with cold color and heat color to offer high attention region for users. It is suitable to highlight interaction region in IER. We render interaction region based on Blignaut's heatmaps algorithm. The set of event *Events* and first snapshot in the set of snapshots collected in the pre-processing stage are used as input. For each event in the event set, the algorithm uses its position as the center point and create a circle with pre-defined radius as interaction region. For each pixel in the interaction region, its color weight is calculated based on Gaussian model. And then, the algorithm computes new RGB for each pixel according to the relation between RGB component composite linear model and weight. The color of points in the interaction region are changed considering both its original color and new RGB.

Algorithm: $\text{RenderInteractionRegion}(Events, P_0, R, W)$

Input: *Events* is event set, P_0 is the first snapshot, R is radius of circle for interaction region, W is weight of color.

Output: P_0 , an image rendered interaction region.

Start

initialize each value of *weightMatrix* as 0

for(*event* in *Events*) {

$centerPoint = event.position$

$Points = \{p | distance(p, centerPoint) \leq R\}$

$weightMatrix[centerPoint] = W$

for(each *p* in *Points*) {

$d = distance(p, centerPoint)$

$coe = e^{-d^2 / (2 \times (0.34 \times R)^2)}$

$weightMatrix[p] = weightMatrix[p] + W * coe \}$

for(each *point* in P_0) {

$w = weightMatrix[point]$

$R = (0 \leq w \leq 50 ? 0 : (50 < w \leq 75 ? 51 * w / 5 - 510 : 255))$

$G = (0 \leq w \leq 25 ? 51 * w / 5 : (25 < w \leq 75 ? 255 : -51 * w / 5 + 1020))$

$B = (0 \leq w \leq 25 ? 255 : (25 < w \leq 50 ? -51 * w / 5 + 510 : 0))$

$temp.R = R, temp.G = G, temp.B = B$

$point.RGB = (point.RGB + temp.RGB) * 0.5 \}$

return P_0

Figure 3 shows the basic content rendered in the interaction region of the IER example using this algorithm. We set parameter R as 82 and W as 100.

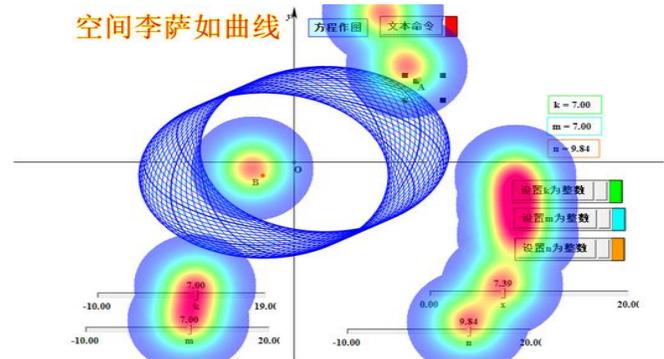


Figure 3. Interaction Region Rendering

2) Dynamic Content Collage

When users interact with IER, the dynamic contents will appear. These dynamic contents can be found out by comparing snapshots. The number of dynamic contents may increase with the complexity of the IER. The dynamic contents may be distributed in different positions. As a result, if we want to offer an effective preview, these dynamic contents are required to be merged into one image.

Collage is a technique to put many small images into one image. It can be used to merge dynamic content of education resource. The first step is selecting representative image. Different from existing collage approach[16] which takes face as important feature and selects important regions by ROI, our approach collects dynamic contents as important regions. The dynamic contents can be found out by comparing snapshots captured during execution. In most cases, a dynamic content

can be considered more important if its area is larger. As a result, our algorithm try to find out the top k dynamic contents with largest area.

Algorithm: GetDynamicContents($P_0, ChangedSnapshotSet, k$)

Input: P_0 is the first snapshot, $ChangedSnapshotSet$ is the set of snapshot except P_0 , k is the number of candidate areas for collage.

Output: $DynamicContents$: the set of the top k dynamic contents with largest area.

computeDifferentArea($snapshot, P_0$) return a rectangle containing dynamic content between $snapshot$ and P_0 .

Start

```

for(each snapshot in ChangedSnapshotSet){
    area=computeDifferentArea(snapshot, P0)
    if(DynamicContents.size < k){
        DynamicContents=DynamicContents ∪ {area}
    }
    else{
        minArea={a|a ∈ DynamicContents ∧
                b ∈ DynamicContents ∧ a<=b}
        if(area > minArea){
            DynamicContents=DynamicContents ∪ {area}-{minArea}
        }
    }
}
return DynamicContents

```

When collaging the set of dynamic contents, the main problem is that the area of dynamic contents are different. For a dynamic content whose area is smaller than cut area, we append it into result regions directly. Otherwise, we use k-Means algorithm to cluster changed areas and select a dynamic content with larger area as main change region. Because the clustering speed is slow when the area of changed region is large, we separate a change region as a set of lines and use midpoint to represent a line during clustering. The midpoints are recovered to lines in changed region when clustering is finished.

The input of the algorithm is the set of dynamic contents $DynamicContents$ generated using the algorithm SelectDynamicContents. In the segment of dynamic content, the content is viewed as a set of horizontal lines. The algorithm is similar when the content is viewed as a set of vertical lines.

Algorithm: CollageDynamicContent($DynamicContents, cutWidth, cutHeight, clusterNum$)

Input: $DynamicContents$ is a set of dynamic contents, $cutWidth$ and $cutHeight$ are used to set the cut area, $clusterNum$ is number of clusters.

Output: An image merged dynamic content.

Start

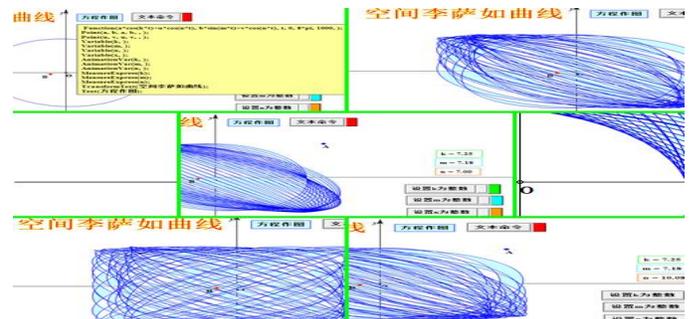
```

for(each content in DynamicContents) {
    if(content.area <= cutWidth × cutHeight)
        Regions.append(content)
    else{
        for(each row in content) {
            lineList= lineList ∪ getLineList(row)
        }
        MidPointSet= {line.midPoint| line ∈ lineList }
        clusterSet = k-Means(MidPointSet, clusterNum)
        maxArea =-1
        for(each cluster in clusterSet){
            lineSet={line| line ∈ lineList ∧ line.midPoint ∈ cluster }
            maxX = max({line.x| line ∈ lineSet })
            minX = min({line.x| line ∈ lineSet })
            maxY = max({line.y| line ∈ lineSet })
            minY = min({line.y| line ∈ lineSet })
            area=(maxX-minX) × (maxY-minY)
            if(area > maxArea){
                maxArea=area
                tempRegion=Rectangle(minX, maxX, MinY, maxY)
            }
        }
        Regions.append(tempRegion)
    }
}
collage Regions into one image as ResultImage

```

return ResultImage

Figure 4 shows the collage of dynamic contents of the example IER “Lissajous-Figur”. We use lines to separate each region. The collage result shows that new elements can be easily found out in the preview. However, there may exist similar contents if a dynamic element moves or the details of an element change according to different operation.



IV. RELATED WORK

Research works on resource preview are developed to save time in interpreting complex contents. Approaches are developed to provide webpage preview, video preview, audio preview and so on. Our work is different that we focus on providing preview for interactive education resources. An interactive education resource opened by its exploring tool is viewed as a knowledge specific application. A lot of techniques such as scaling, pre-fetch and precomputation, collage, key information extraction, resource visualization are developed in the preview generation approaches. Our approach also employs techniques used in these works .

In order to generate preview for web page, Yoo et al[3] use thumbnail to offer link preview. They adjust thumbnails' size and transparency according to the distance between cursor and link. Aula et al [4] compare visual web page preview with textual web page preview on predicting web page's benefit and find that adding title and url around thumbnail can decrease underestimated prejudice. Wittenburg et al[7] regard outer links as potential content user's looking for and organize images of outer links as image streaming to generate image preview.

In order to generate preview for complex interaction, Rekimoto[6] designs "PreSense", which is a touch interactive press key device. Users can slide key and know what will happen if users press this key. Drewes et al [5] use translucent bitmap to enhance tool tips, they override the function "GetMessageString" in the framework under windows platform to implement that cursor moves on window menu widget will show translucent dialog.

In order to generate preview for video, Barletta et al [9] extract video, audio, text key information from video, offer 2~3 minutes main key frames and second key frames under 20 seconds for users. Craggs et al [10] propose a approach to record user's tags for video frame when users are watching videos. A server uses meta-data and tag to return a video animation which made up of key frames. Zhao[17] offers visualization for open education resource video. Their approach extracts key frames from video and uses collage technique to organize important content into one image. Komlodi et al[18] use dynamic and static key frames to offer the preview of video.

Audio preview is also used in the context that users can not look at the message directly. Parente et al [11] develop AEI software to offer preview of audio files for blind users. Their approach finds document hyperlink and extracts content to read for blinders. Shirazi et al [12] extract SMS content, analyze the content to get SMS type and play the tune corresponding SMS type for users.

A lot of techniques are developed in the preview generation approaches. Scaling[3,4,18], pre-fetch and precomputation [5,6,7,11,12], key information extraction[9,10,17] and collage[16,20] are widely used to generate preview. Ahn et al[8] use icon's brightness and darkness degree to analyze the relativity which multiple documents to the same theme. Khan et al [19] use CSG to extract concept of course and use concept node to form course map. Luo et al[20] propose a way to collage important video content in one image.

V. CASE STUDY

A. Case Study Setting

We use our approach to generate preview of interactive education resource for Super Sketchpad, which is a software for teaching mathematics by providing geometry drawing and reasoning during creation of education resource. The preview functionality is implemented in the resource search engine and providing service to students learning courses related to Super Sketchpad in Guangzhou University (in the Website: <http://www.appoperation.com/search/>). The search engine now contains about 1,000 interactive education resources of the type Super Sketchpad. We randomly select 20 interactive education resource which have at least 5 events to valid our approach.

15 students from Sun Yat-sen University are invited to evaluate preview effect. Each participant chooses to evaluate 10-20 preview. They are asked to download the tool of Super Sketchpad and the resources, and then use the tool to explore and interact with the resource manually. At last, they compare the preview with their understanding of the resource. For each resource, participants are asked to evaluate the completeness, correctness, and comprehension ability of the preview and give a score.

In this paper, preview completeness means resource information offered by the preview compared to the entire information in resource; preview correctness indicates the extent users are able to understand content in resource correctly without ambiguity; preview comprehension means comprehension easiness or difficulty users understand content in resource by the preview. For each preview dimension, score for rating can be from 1 to 5. Higher score means better preview effect on preview dimension. In addition, participants are asked to answer whether the preview save time, how much time is saved and how much content can be understood from the preview.

B. Experiment Result

Figure 5 shows the preview rating on three valid dimensions of 15 participants: completeness, correctness, easy comprehension. The vertical axis shows the rating scores. The higher preview rating is, the better the preview effect is. The horizontal axis shows the name of resource used in the evaluation. According the experiment result, the average score for preview completeness is 4.42, the average score for preview correctness is 4.53, the average score for preview comprehension is 4.28. It indicates that the preview include most key contents and better summarized preview effect for IER.

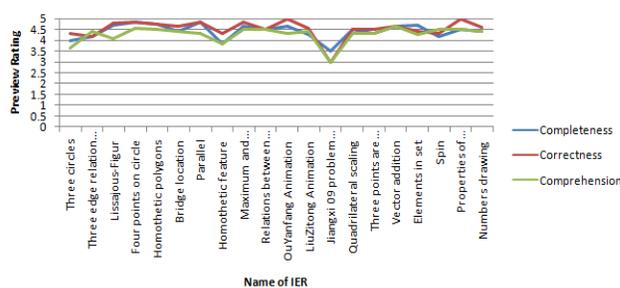


Figure 5. Rating of IER Preview

All participants agree that using the preview can save time during searching resource of Super Sketchpad. Figure 6 shows the percentage of time saved using our preview. According to the experiment result, users can save 54.89% of time on average using the preview. It saves half the time instead of user's complex manual exploration. Users have 69.04% content understanding on average. It means that users can get general content from IER and don't need to download IER.

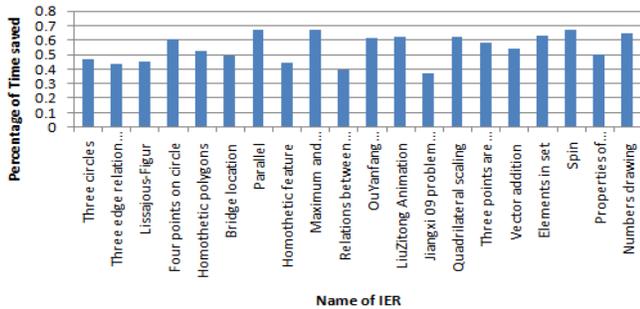


Figure 6. Percentage of Time Saved

VI. CONCLUSIONS AND FUTURE WORK

In this paper, we propose a summarized preview approach based on exploring and comparing GUIs for interactive education resource. The summarized preview is an image included basic content, interaction region and dynamic content. The experiment result shows that the preview can be used to provide most information in the resource correctly and easy to understand.

In this paper, we implement our approach for the IERs developed using Super Sketchpad. In our future work, we will apply our approach in different kinds of IERs to evaluate the extensibility and effect of our approach in generating preview for education resources.

ACKNOWLEDGMENT

This research is supported by the NSFC Guangdong Joint Fund (No. U1201252), the Science and Technology Planning Project of Guangdong Province (No. 2014B010110003), National Natural Science Foundation of China (No. 61502546, 61370186), and National High-tech R&D Program (863 Program) (NO. 2015AA015408).

REFERENCES

- [1] Zeina Atrash Leong, Michael S. Horn. The BEAM: a Digitally Enhanced Balance Beam for Mathematics Education. In IDC '10 Proceedings of the 9th International Conference on Interaction Design and Children, New York: ACM, 2010, 290-292
- [2] Oai Ha, Ning Fang. Development of Interactive 3D Tangible Models as Teaching Aids to Improve Students' Spatial Ability in STEM Education, In Frontiers in Education Conference, IEEE, 2013, 1302-1304
- [3] ByungIn Yoo, JongHo Lea, YeunBae Kim, The seamless browser: enhancing the speed of web browsing by zooming and preview

- thumbnails, WWW '08 Proceedings of the 17th international conference on World Wide Web, New York, ACM, 2008, 1019~1020
- [4] Aula A, Khan R M, Guan Z, et al, A comparison of visual and textual page previews in judging the helpfulness of web pages, In Proc. WWW '10 Proceedings of the 19th international conference on World wide web, New York, ACM, 2010, 51~60
- [5] Heiko Drewes, Albrecht Schmidt, WYSIWYG-Tool Tips: Enhancing Tool Tips with Translucent Preview Bitmaps, Advances in Visual Computing Volume 3804 of the series Lecture Notes in Computer Science, USA, 647~652
- [6] Jun Rekimoto, Takaaki Ishizawa, Carsten Schwesig, et al, PreSense: interaction techniques for finger sensing input devices, UIST '03 Proceedings of the 16th annual ACM symposium on User interface software and technology, New York, ACM, 2003, 203~212
- [7] Kent Wittenburg, Wissam Ali-Ahrnad, Daniel LaLiberte, et al, Rapid-Fire image Previews for information Navigation, In AVI '98 Proceedings of the working conference on Advanced visual interfaces, New York, ACM, 1998, 76~82
- [8] Jae-wook Ahn, Peter Brusilovsky, Guiding Educational Resources for iSchool Students with Topic-based Adaptive Visualization, Proceedings of the 2011 iConference, New York, ACM, 2011, 632~633
- [9] A. Barletta, M. Mayer, B. Moser, Leafing digital content, IUI '04 Proceedings of the 9th international conference on Intelligent user interfaces, New York, ACM, 2004, 217~219
- [10] Barnaby Craggs, Myles Kilgallon Scott, Jason Alexander, ThumbReels: Query-Sensitive Web Video Previews Based on Temporal, Crowdsourced, Semantic Tagging, CHI '14 Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, New York, ACM, 2014, 1217~1220
- [11] Parente P, Audio Enriched Links: Web Page Previews for Blind Users, In Proceedings of the 6th international ACM SIGACCESS conference on Computers and accessibility, New York, ACM, 2003, 2~8
- [12] Alireza Sahami Shirazi, Ari-Heikki Sarjanoja, Florian Alt, et al, Understanding the Impact of Abstracted Audio Preview of SMS, In CHI '10 Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, New York, ACM, 2010, 1735~1738
- [13] AutoHotKey .https://autohotkey.com/
- [14] Daniela Chudá Visualization in education of theoretical computer science, CompSysTech '07 Proceedings of the 2007 international conference on Computer systems and technologies, New York, ACM, 2007, Article No. 84
- [15] Pieter Blihnaut, Visual span and other parameters for the generation of heatmaps, ETRA'10 Proceedings of the 2010 Symposium on Eye-Tracking Research & Applications, New York, ACM, 2010, 125~128
- [16] Rother C, Bordeaux L, Hamadi Y, et al, Autocollage, SIGGRAPH '06 ACM SIGGRAPH 2006 Papers, New York, ACM, 2006, 847~852
- [17] Baoquan Zhao, Songhua Xu, Shujin Lin, et al, A new visual navigation system for exploring biomedical Open Educational Resource (OER) videos, Journal of the American Medical Informatics Association Advance Access, 2015, pii: ocv123. doi: 10.1093/jamia/ocv123
- [18] Anita Komlodi, Gary Marchionini, Key Frame Preview Techniques for Video Browsing, In Proc. DL'98, New York, ACM Press, 1998, 118~125
- [19] Javed I. Khan, Manas S. Hardas, Observing Knowledge Clustering for Educational, In K-CAP '07 Proceedings of the 4th international conference on Knowledge, New York, ACM, 2007, 193~194
- [20] Sheng-Jie Luo, Chun-Yu Tsai, Hsiao-Ching You, et al, Previewing video content with dynamic and interactable collage, SA '12 SIGGRAPH Asia 2012 Posters, New York, ACM, 2012, Article No. 21
- [21] Jingzhong Zhang, Xichdeng Peng. Free Software SSP for Teaching Mathematics. Symbolic Computation and Education. 2007.8. pp:115-135

Learning to Discover Subsumptions between Software Engineering Concepts in Wikipedia

Xiang Dong, Kai Chen, Jiangang Zhu, Beijun Shen
School of Electronic Information and Electrical Engineering

Shanghai Jiao Tong University, Shanghai, China

Email: dongxiang@sjtu.edu.cn, voyageckg@sjtu.edu.cn, jszjgws@sjtu.edu.cn, bjshen@sjtu.edu.cn

Abstract—Wikipedia contains large-scale concepts and rich semantic information. A number of knowledge base construction projects such as WikiTaxonomy, DBpedia, and YAGO have acquired data from Wikipedia. Despite the huge amount of relations in Wikipedia, the semantic relations (i.e. subsumptions) between domain concepts are rather sparse, especially in software engineering (SE) area. Hence, it is difficult to derive a software engineering knowledge base directly from Wikipedia. Meanwhile, domain knowledge base has become indispensable to a growing number of applications in software engineering. So the discovery of missing semantic relations between software engineering concepts in Wikipedia is essential. In this paper, we propose an approach to automatically discovering the missing subsumption relations between software engineering concepts. Specifically, we extract the SE domain concepts from Wikipedia firstly. And secondly, we design a machine learning based algorithm with some novel features to calculate the semantic relevancy between concepts. Thirdly, we offer and utilize a semi-supervised model to incorporate the features, which discovers the SE subsumptions. Experimental results show that our approach can effectively find the missing subsumption relations between software engineering concepts. Finally, we build a taxonomy which contains 193,593 concepts together with 357,662 subsumption relations. Compared with the taxonomies which are extracted from general-purpose knowledge bases such as WikiTaxonomy, YAGO and Schema.org, our dataset has a larger scale in software engineering domain.

Index Terms—Subsumption Extraction, Software Engineering, Wikipedia

I. INTRODUCTION

Wikipedia is the largest online encyclopedia in the world. It contains more than 5,027,000 concepts, and its extensive network of links, categories and infoboxes provide a variety of explicitly defined semantics. Thus, some knowledge bases use Wikipedia as the data source of relation extraction. For example, Wikitaxonomy [1], DBpedia [2], and YAGO [3] are the large-scale knowledge bases which are derived from Wikipedia.

However, these general-purpose knowledge bases only contain a small quantity of concepts and relations in software engineering (SE) domain. There is no one owning more than 1,000 SE concepts or relations. For instance, the subsumption between “Sandbox (computer security)” and “Virtualization software” is a subsumption relation in software engineering, and “Sandbox” is an important SE concept, but they can not be found in existing knowledge bases. Especially, a lot

of relations between SE concepts are not discovered when analyzing knowledge data from Wikipedia. However, they are more important for SE researches and practices such as semantic relatedness calculation [4] [5] [6], document correlation analysis [7], and defect prediction.

In this paper, we propose an approach to automatically discovering the SE domain subsumptions in Wikipedia, so that the missing SE semantic subsumption can be gathered and established. However, to complete the work, we would face these challenges:

- Software engineering concepts are always composed of domain-specific terms. While natural language processing techniques like segmentation or pos tagging are the basis of some Web-based approaches for taxonomy construction, directly applying these approaches to our scenario will lead to poor results.
- Several structural information is contained in Wikipedia. How to design an algorithm to incorporate this information when detecting subsumptions between concepts is another important topic.
- The invalid SE entities would increase the difficulty of subsumptions prediction when taking the Wikipedia structural information into consideration.

To solve the challenges above, we propose a machine learning based approach for domain subsumption prediction. Particularly, it leverages several features from different aspects to measure the semantic relatedness between concepts. Thus, the confidence of relation prediction would be increased. The semi-supervised model has been proposed to process the information. We split the whole machine learning process into several iterations to extract and optimize the relations set. So in this paper, our work and contributions mainly include:

A. Using wiki-based features for classifiers training: we utilize the structure based features in Wikipedia, and combine them with the lexical, co-occurrence and distribution features to calculate similarity among different concepts. It optimizes the performance of subsumption extraction.

B. Automatic labeling and iterating: we set the constraint-based rules to label the negative data, and adopt subsumption patterns to extract the positive data. During each iteration of the semi-supervised process, several constraints are defined to automatically optimize the relations set, and in this way, we shrink the cycle of data handling.

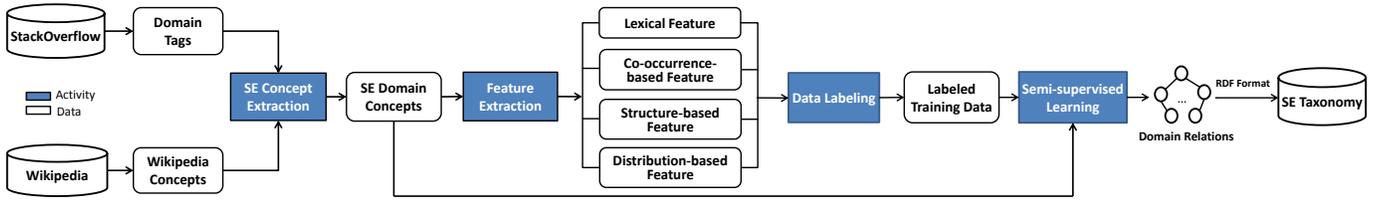


Fig. 1: Our approach for extracting SE relations in Wikipedia

C. *Extracting SE subsumptions and building taxonomy:* we discover and extract the SE subsumptions in Wikipedia, and build the large-scale taxonomy in software engineering domain. The experimental results show our taxonomy contains the large-scale concepts and deep semantic hierarchies.

II. RELATED WORK

For using Wikipedia, there have been a large amount of researches on relations extraction in general area. Auer et al. presented the DBpedia system [2] which generates RDF statements by extracting the attribute-value pairs contained in infoboxes of Wikipedia (e.g. country = [[the People’s Republic of China]] is a key-value pair in ShangHai page). Suchanek et al. built the YAGO system which refers to relation exploration [3]. The method of YAGO construction is getting all the “is-a” hierarchy in WordNet and then mapping into Wikipedia for corresponding instances. Merging Wikipedia with WordNet makes YAGO contain over 200,000 classes and 400,000 subsumption relations. Weld et al. in Kylin Ontology Generator (KOG) [8] built a subsumption hierarchy by combining Wikipedia infoboxes with WordNet using statistical-relational learning. WikiTaxonomy [1] used the way of finding some special rules for subsumption in Wikipedia category structures. For example, “Companies listed on NASDAQ” is a category of “Microsoft” and it is the plural format, so WikiTaxonomy refers the pair as a subsumption relation. Furthermore, WikiTaxonomy uses inference method to extract the relationship again based on the existing concepts, and it totally obtains 105,000 concepts with the precision of 88%. Zhishime [9] is the trial for relations discovery based on Chinese Wikipedia. In addition, Lin et al. [10] used Freebase as the supplement of Wikipedia, finding the relations by solving invalid entities in Wikipedia. The BabelNet [11] uses a semi-automatic approach to gather all the relations mined from Wikipedia, WordNet and the results of Google machine translation. Wu et al. presented a special method to construct probase [12] system by assigning the probabilities to each relationship. It extracts 2.7 million concepts.

As for relations discovery in SE domain, Lextcal Views [13] applied some natural language processing techniques to automatically extract and organize concepts relationship among software identifiers in a WordNet-like structure. But more software programming terms would not be included in this relation set because Lextcal Views only use the software identifiers as its input. Zhu et al. [14] extracted hypernym-hyponym relations between tags in StackOverflow, and built

a taxonomy called Software.zhishi.schema. It obtains 38,205 concepts and 68,098 relations, which still has the rise of space in scale.

III. APPROACH

A. Approach Overview

In order to discover the subsumption relations between the software engineering concepts, a machine learning based approach is proposed, as shown in Fig 1. It consists of four key activities:

1) *SE Concept Extraction:* through the input of SE domain tags in StackOverflow and Wikipedia concepts, we extract the SE domain concepts as preparation of the relations discovery.

2) *Feature Extraction:* features are launched for machine learning, and these are divided into four classes: *Lexical Features*, *Co-occurrence-based Features*, *Structure-based Features* and *Distribution-based Features*.

3) *Data Labeling:* it generates the labeled data for classifiers. For negative data, we use constrains-based method for training. And for positive data, Hearst [15] patterns are adopted to get some subsumptions for positive data generation.

4) *Semi-supervised Learning:* the semi-supervised training process consists of several iterations. And in each iteration, we review the trained results of the current iteration and update high confidence data as the input of the next.

Finally, it extracts 357,662 subsumptions in our relation set, and based on the discovered subsumptions, we build the SE domain taxonomy in RDF format. We deploy our taxonomy on datahub website (<https://datahub.io/dataset/setaxonomy>). Following subsections will describe these activities in detail.

B. Domain Extraction

The process of domain extraction can be illustrated as following steps: first, we get the domain tags from StackOverflow, and it contains tens of thousands of items. Through the word-embedding method and voting mechanism, we filter out all the domain-irrelevant tags and preserve the seed words which are needed. Second, in order to get structural information from Wikipedia, we dump the Wikipedia XML file (<https://dumps.wikimedia.org/enwiki/20150901/>) and obtain a large-scale set which contains 5,027,000+ concepts. Third, based on the built concept repository, we extract the domain concepts by using seed words derived from the StackOverflow. As the result, we obtain the software engineering corpora containing 193,593 concepts, which is for the later relations discovery process.

C. Feature Extraction

We adopt classification models to determine the correct subsumption relation of each concept pair, for which several effective features are designed. The purpose of feature engineering is to quantitatively characterize the similarities or relatedness between concepts. We divide all the features into four classes: *Basic Features*, *Co-occurrence-based Features*, *Structure-based Features* and *Distribution-based Features*.

1) **Basic Features**: this type of feature aims at capturing the lexical similarity between two concepts, and it consists of two features: *Head correlation calculation* and *Unbalanced common substring*.

Feature 1. Head correlation calculation: “head” means the core word of concept. As for the analysis of head matching, our approach is to build the semantic tree by using Stanford Parser [16]. We judge and extract the core noun phrase from the tree as the head of the term. After obtaining the subsumption’s head, we adopt *WUP* [17], which is the similarity calculation method based on WordNet structure, to measure the relatedness. The *WUP* formula is as follows:

$$WUP(X, Y) = \frac{2 \cdot \text{depth}(LCA(H_x, H_y))}{\text{depth}(H_x) + \text{depth}(H_y)} \quad (1)$$

where H_x, H_y are the head of the concepts X and Y . LCA is the lowest common ancestor of H_x and H_y in WordNet.

However, if there exists several heads in one concept, we compute *WUP* for all, and then select the maximum value. For example, as for concept “Software using the Apache License” and “Android (operating system)”, we fetch the heads “Software” and “Apache license” in first concept, and fetch the “Android” in second one. Then, we calculate the *WUP* values both between “Software” & “Android” and “Apache license” & “Android”. The maximum of these two values is served as the final feature figure of the relation between “Software using the Apache license” and “Android (operating system)”.

Feature 2. Unbalanced common substring: this feature is designed to calculate the longest common substring of two concepts, and we get the unbalanced lexical length character into formulating construction:

$$f(X, Y) = \frac{LCS(X, Y)}{Len(X)} \quad (2)$$

where $Len(X)$ means the length of X , and $LCS(X, Y)$ means the longest common substring of X and Y . Increase of the value means X is more likely to be the hypernym of Y .

2) **Co-occurrence-based Features**: some relations hold the subsumptions with low lexical similarities. Thus, we leverage the co-occurrence information, which is in the content of the concept, to measure the semantic relatedness between two concepts.

For the co-occurrence-based features, we first define the formula of co-occurrence calculation as follows:

$$r(a, b) = \frac{\log(\max(|I_a|, |I_b|)) - \log(|I_a \cap I_b|)}{\log(|W|) - \log(\min(|I_a|, |I_b|))} \quad (3)$$

The formula represents Normalized Google Distance (NGD), where a and b are the concepts of Wikipedia, and I_a, I_b are the inner-link sets in article. Feature 3 and Feature 4 are defined based on this formula.

Feature 3. Abstract co-occurrence calculation: abstract is a brief introduction, and contains the concept’s definition. So abstract can effectively reveal the main characteristics of current concept.

$$f_{abs}(a, b) = r(I_{abs_a}, I_{abs_b}) \quad (4)$$

We define the abstract co-occurrence calculation feature $f_{abs}(a, b)$ above, and I_{abs_a}, I_{abs_b} are the link sets in abstract structure of a and b .

Feature 4. Text co-occurrence calculation: text is the main text body of concept, and it is responsible for comprehensively describing the concept with enough related information. We define the co-occurrence calculation in text:

$$f_{txt}(a, b) = r(I_{txt_a}, I_{txt_b}) \quad (5)$$

where I_{txt_a}, I_{txt_b} are the link sets in main text body of concepts.

Feature 5. Category co-occurrence calculation: the category lists the concepts which current concept belonged to, so some potential subsumptions are contained in the category. In this feature, we calculate the co-occurrence in category structure. Furthermore, if a is in the category of b or vice versa, they may be more likely a subsumption, so we add weight to this situation based on *NGD* formula. The $f_{cate}(a, b)$ is as follows:

$$\frac{\log(\max(|I_a|, |I_b|)) - \log(|I_a \cap I_b| + f_{bls}(a, b))}{\log(|W|) - \log(\min(|I_a|, |I_b|))} \quad (6)$$

In the $f_{cate}(a, b)$ above, $f_{bls}(a, b)$ represents the weight of inclusion. We define $f_{bls}(a, b)$ in equation (7), and $\mu, \nu > 0$:

$$f_{bls}(a, b) = \begin{cases} \mu & , a \in I_{cat_b} \\ \nu & , b \in I_{cat_a} \\ 0 & , \text{others} \end{cases} \quad (7)$$

where I_{cat_a} and I_{cat_b} are category sets of a, b . Based on the analysis of weight impact, we set $\mu = 5, \nu = 5$ in our experiment.

3) **Structure-based Features**: the *Co-occurrence-based Features* are only calculated for relevancy in the content, but the analysis in Wikipedia structure is also important. Thus, we launch the *Structure-based Features*.

Feature 6. Similarity calculation of guideline: guideline is a Wikipedia structure, and it lists the chapters which would be illustrated in the text body of page. For example, in page “Binary search Tree”, the guideline includes chapters “Definition”, “Operations” and “Examples of applications” etc., and the “Operations” also includes “Searching”, “Insertion” and “Deletion”. These listed chapters are mapped to the text body and described in detail. Thus, guideline reveals the whole structure of the text, and it is also one of the

important structures. We use *Jaccard* to calculate the similarity in guideline.

$$Jaccard(I_{gdl_a}, I_{gdl_b}) = \frac{I_{gdl_a} \cap I_{gdl_b}}{I_{gdl_a} \cup I_{gdl_b}} \quad (8)$$

where I_{gdl_a}, I_{gdl_b} are the chapter sets of guidelines.

Feature 7. Similarity calculation of infobox: infobox is another structure provided by Wikipedia, and it lists the attributes to present the information of concept. We take I_{info_a}, I_{info_b} into formula (8) to calculate the *Jaccard* value of infobox, and the I_{info_a} and I_{info_b} mean the attribute sets of infoboxes.

4) **Distribution-based Features:** this feature is for measuring the difference between two topic distributions of two different concepts.

Feature 8. Divergence calculations: as for the subsumptions that have low similarity in content and structure, we adopt *KL-divergence* [18] to do the further analysis. In the process of *KL-divergence*, we use *LDA* [19] to build the Wikipedia topic-based distribution, and calculate the probability distributions of the two concepts. Finally, we compute the *KL-divergence* between the two distributions.

$$D_{KL}(p_{wa} || p_{wb}) = \sum_{n=1}^N p_{wa}(n) \log \frac{p_{wa}(n)}{p_{wb}(n)} \quad (9)$$

where $p_{wa}(n)$ and $p_{wb}(n)$ are the probability of n -th topic in the topic distributions of a, b .

D. Data Labeling

Subsumption detection is usually treated as a binary class classification problem. And classification is a supervised learning, which requires labeled data for training. Thus, the generation activity should be taken into consideration for training adequate and fine distribution labeled data.

As for the positive data, we use Hearst patterns in abstract of each page to find some subsumptions as our positive data. The following patterns are used in our approach:

NP1{,} “such as” NPList2
 NP1 {,} “and other” NP2
 NP1 {,} “including” NPList2
 NP1 “is a” NP2
 NP1 “is the” NP2 “of” NP3

Part of subsumptions are extracted by Hearst, and we will select some subsumptions and calculate the feature values for them, to label as the positive data.

As for the negative data, we first extract some relations between concepts, and calculate the feature values. Through feature-based constraints, we predict if these relations meet the negative conditions. For example, about the undetermined relation (X, Y) , we proceed the judgment on following constraints:

- 1) $WUP(X, Y) < M$
- 2) $Len(X) > Len(Y)$
- 3) $Jaccard_{info} = 0$

- 4) $Jaccard_{gdl} = 0$
- 5) $|KL(X, Y) - KL(Y, X)| < N$

where $WUP(X, Y)$ calculates the *WUP* value between X and Y , $Len(X)$ gets the length of X , and $KL(X, Y)$ obtains the *KL-divergence* between X and Y . In these constraints, M and N are set as ascertained value to meet the requirement of our engineering. If any constraint above is satisfied, relation (X, Y) will be labeled as negative. In our experiment, $M = 0.4$ and $N = 0.003$.

E. Semi-supervised Learning

We design a semi-supervised learning algorithm to extract SE relations, which splits the whole training process into several iterations. In the first iteration, we import the labeled training data and SE domain concepts. And in the following iteration, the input is the optimized results processed by the last iteration. We totally enforce 5 iterations and build a SE domain relation set as outcome.

However, the data quality in previous iteration could make a huge impact on the next iteration, therefore some incorrect data if not discarded may spread along the progress of iteration and cause more inaccuracy. In order to avoid this mistake, we define some rules between two adjacent iterations to filter out the incorrect relations. In this paper, we launch three useful constraints as follows:

- **Constraint 1:** ring conflict constraint. Given the two concepts, they link to each other will be not allowed. This is because the subsumed relation is asymmetric. For example, If we find a subsumption “Sorting algorithms” \rightarrow “Quicksort”, and then find “Quicksort” \rightarrow “Sorting algorithms”, the relation “Quicksort” \rightarrow “Sorting algorithms” should be removed.
- **Constraint 2:** transitive redundancy constraint. Given concepts a, b and c . If a subsumes b , and b subsumes c , then a must subsume c since the subsumption satisfies transitivity. This subsumption is non-essential in our relation set. For instance, the concept “word2vec” subsumed into “machine learning” is unnecessary because “word2vec” is subsumed by “deep learning” and “deep learning” is subsumed by “machine learning”.
- **Constraint 3:** synonymous conflict constraint. if concept a and concept b are synonymous, they must not be subsumptive. Therefore, such relations like “Ordered binary tree” \rightarrow “Binary search tree”, or “Heap” \rightarrow “Heap” would be removed in this constraint.

These three constraints are very important for ensuring the fine quality of the data output in each iteration, because constraint 1 and 2 contribute to avoiding pulling in the incorrect instance, and constraint 3 helps cast off the redundancy. This mechanism optimizes the result and guarantees our approach to learn more relations with fine granularity.

IV. EXPERIMENT

A. Experiment Setup

1) **Data handling:** some necessary data are processed in our approach. During the data extraction in SE concept extraction

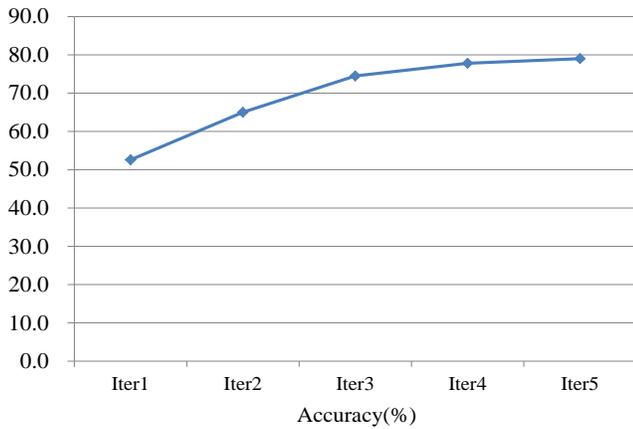


Fig. 2: Accuracy tendency for each iteration

TABLE I: Result sets of three models

Models	Relation numbers	Accuracy
Decision Tree	333,659	77.3%
Naive Bayes	351,457	78.1%
SVM	357,662	79.7%

activity, we totally obtain 193,593 software engineering domain concepts. And in the data labeling activity, we finally generate 4,181 labeled training data, which includes 2,018 positive data and 2,163 negative data, and these labeled data are used for training the classifiers.

2) *Accuracy evaluation*: as for the accuracy evaluation in our experiment, we mainly use manual judgment to finish this work. For the classification results in each iteration, we invite 5 partners in our laboratory to join the accuracy calculation work. We randomly select about 10,000 relations extracted by the certain classifier in each iteration, and assign approximately 2,000 relations to each member. We require them to judge the exactness of these relations by labeling “Yes”, “No” or “Uncertain” for each relation. After finishing each part of works, we gather all the results and estimate the prediction accuracy of classifiers.

3) *Semi-supervised learning*: for assuring the stability of extraction in semi-supervised, we implement three models for information processing, which are decision trees, naive Bayes and SVM. These models extract relations and get three different result sets, and we select the best one as our final result and put it into taxonomy with the RDF format.

B. Result Analysis

1) *Accuracy evaluation of semi-supervised learning*: we do the accuracy evaluation of semi-supervised learning in order to check the effect of several iterations and the constraints mentioned in III(E). As for the used three classifiers, we separately calculate the accuracy of each algorithm. And in every iteration, we get the average accuracy of three models at current iteration. In this way, we ensure the stability of evaluation. The tendency chart of precision is Fig 2.

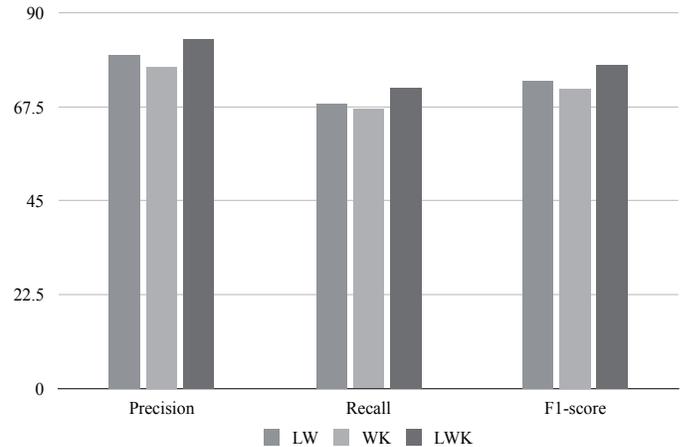


Fig. 3: Comparison of different feature groups

2) *Results analysis of three models*: after the whole semi-supervised process is finished, the final results extracted by three models are shown in Table I. We could find the results trained by SVM is the best, and we use SVM results as our final relationship set which contains 357,662 subsumptions with 79.7% accuracy.

C. Feature Contribution Analysis

We design the experiment to analyze whether these features are helpful for the relationship prediction. In the experiment, all the features first are divided into three classes: *Lexical Features*, *Wiki-based Features*(including *Co-occurrence-based Features* and *Structure-based Features*), and *KL-Divergence Feature*. Then, we combine these classes into three groups. The first group consists of *Lexical Features* and *Wiki-based Features*(donated as *LW*), and the second group includes *Wiki-based Features* and *KL-Divergence Feature*(donated as *WK*), for the third group, we use all the features(donated as *LWK*). As for the three groups, we trained the three SVM classifiers separately and get three different results. The precision, recall and F1-scores are calculated for the three result sets and we make comparisons in Fig 3.

The result shows *LWK* performs best, the values of precision, recall and F1-score are higher than *LW* and *WK*, which means all the features have an effect on subsumption extraction. Compared with *WK*, the *LW* performs better. It is mainly because *Lexical Features* is adept at finding overt subsumptions which contain semantics relations, and the *Lexical Features* could predict the relation when *Wiki-based Features* are symmetry. On the contrary, *KL-Divergence Feature* aims at finding the cryptic subsumptions, which could be small amount and lower confidence compared with *LW*.

D. Taxonomy Comparison

Since the absence of public software engineering taxonomy, we first take the software engineering taxonomy built by Zhu et al. [14] into consideration, the result set is called *Software.zhishi.schema*. Some general knowledge bases are also imported into our experiment, such as *Yago Taxonomy*,

TABLE II: Comparison with other works

	Ours	Software. zhishi.schema	SE subset of		
			YAGO	WikiTaxonomy	Schema.org
Concept Number	193,593	38,205	898	711	10
Subsumption Number	357,662	68,098	870	630	0
Avg # of Parents	1.5	1.1	0.9	0.86	0
Avg # of Children	2.21	1.85	0.73	0.71	0
Maximun Depth	31	28	3	6	1
Minimum Depth	1	1	2	1	1
Average Depth	7.02	6.99	2.24	1.39	1.00

WikiTaxonomy, and Schema.org. We compare the SE domain subset of above works, and provide the indicators such as concept number, subsumption number, maximum depth, minimum depth and average depth for each collection in order to illustrate the granularity and richness.

From the Table II, we can see in our taxonomy, the number of concepts is 193,593 and the number of subsumptions reaches 357,662, which contains larger number of concepts and subsumptions than other works.

TABLE III: Some classifications and their typical instances

Classifications	Num	Typical Instances
Operating Systems	97	Solaris, Linux, Microsoft Windows, OS X
Web Browsers	171	Firefox, Safari, Internet Explorer, Chrome
Internet Protocols	235	HTTP, FTP, SMTP, GNTTP
Sorting Algorithms	196	Heapsort, Quicksort, Radix Sort, Selection Sort
Deep Learning Software	103	CNTK, Caffe, OpenNN, Torch
Classification Algorithms	110	Random Forest, AdaBoost, ID3 Algorithm, Decision Tree

We randomly select six classifications in our taxonomy as shown in Table III. The experiment is designed for demonstrating the richness and semantic correctness of classifications in detail. Table III. shows that the selected six classifications above contain plenty of concepts which cover almost all the conceptions of the corresponding subareas. And through the random sampling, 95% of the classifications in our datasets keep the same condition. Thus, our dataset is proved to possess the abundant and precise SE concepts. The complete taxonomy has been deployed on datahub website.

V. CONCLUSION

In this paper, we propose an approach that discovers the subsumptions between SE concepts from Wikipedia. The approach collects domain tags in StackOverflow as seed words, extracts concepts in Wikipedia, and uses machine learning algorithms to extract subsumption relations. We launch multi-dimension features to improve the training precision. As a result, we build the taxonomy which contains 193,593 concepts and 357,662 subsumption relations with the format of RDF.

The experimental results demonstrate the large-scale and high accuracy of our dataset.

For the future work, we will try to use other datasets to support the current work, because we find there is still increasing space of scale if the entity invalidations of Wikipedia can be thoroughly solved. It requires implementing the mapping mechanism of entities between Wikipedia with other datasets.

VI. ACKNOWLEDGEMENT

Beijun Shen is the corresponding author. And this research is supported by 973 Program in China (Grant No. 2015CB352203) and National Natural Science Foundation of China (Grant No. 61472242).

REFERENCES

- [1] Simone Paolo Ponzetto, Michael Strube:WikiTaxonomy: A Large Scale Knowledge Resource. ECAI 2008: 751-752.
- [2] Soren Auer, Christian Bizer, Jens Lehmann, Georgi Kobilarov, Richard Cyganiak, and Zachary Ives, DBpedia: A nucleus for a Web of open data, in Proc. of ISWC 2007 + ASWC 2007, 722735, (2007).
- [3] Johannes Hoffart, Fabian M Suchanek, Klaus Berberich, and Gerhard Weikum. Yago2: a spatially and temporally enhanced knowledge base from wikipedia. Artificial Intelligence, pages 163,(2012).
- [4] Michael Strube, Simone Paolo Ponzetto: WikiRelate! Computing Semantic Relatedness Using Wikipedia. AAAI 2006: 1419-1424.
- [5] Budanitsky, A. & G.Hirst. Evaluating WordNet-based measures of semantic distance. Computational Linguistics 2006, 32(1).
- [6] Giriprasad Sridhara, Emily Hill, Lori Pollock, and K Vijay-Shanker. Identifying word relations in software: A comparative study of semantic similarity tools. In ICPC 2008, pages 123132. IEEE, 2008.
- [7] Rongxin Wu, Hongyu Zhang, Sunghun Kim, and Shing-Chi Cheung. Relink: recovering links between bugs and changes. In Proceedings of the 19th ACM SIGSOFT symposium and the 13th European conference on Foundations of software engineering, pages 1525. ACM, (2011).
- [8] Fei Wu and Daniel Weld, Automatically refining the Wikipedia infobox ontology, in Proc. of WWW-08, (2008).
- [9] Xing Niu, Xinruo Sun, Haofen Wang, Shu Rong, Guilin Qi, Yong Yu: Zhishi.me - Weaving Chinese Linking Open Data. International Semantic Web Conference (2) 2011: 205-220
- [10] Thomas Lin, Mausam, and Oren Etzioni. No noun phrase left behind: Detecting and typing unlinkable entities. In Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, pages 893-903, Jeju Island, Korea, July.
- [11] Navigli R, Ponzetto S P. BabelNet: The automatic construction, evaluation and application of a wide-coverage multilingual semantic network. Artificial Intelligence, 2012, 193: 217-250.
- [12] Wentao Wu, Hongsong Li, Haixun Wang, and Kenny Q Zhu. Probase: A probabilistic taxonomy for text understanding. In SIGMOD 2012, pages 481-492. ACM, 2012.
- [13] J-R Falleri, Marianne Huchard, Mathieu Lafourcade, Clementine Nebut, Violaine Prince, and Michel Dao. Automatic extraction of a wordnet-like identifier network from software. In Program Comprehension (ICPC), 2010 IEEE 18th International Conference on, pages 413. IEEE, (2010).
- [14] Jiangang Zhu, Beijun Shen, Xuyang Cai, Haofen Wang:Building a Large-scale Software Programming Taxonomy from Stackoverflow. SEKE 2015: 391-396
- [15] Marti A Hearst. Automatic acquisition of hyponyms from large text corpora. In Proceedings of the 14th conference on Computational linguistics-Volume 2, pages 539-545,(1992).
- [16] Dan Klein and Christopher D. Manning. Accurate unlexicalized parsing. In Proceedings of the ACL, (2003).
- [17] Wu Z, Palmer M. Verbs semantics and lexical selection. Proceedings of the 32nd annual meeting on Association for Computational Linguistics. Association for Computational Linguistics, 1994: 133-138.
- [18] Solomon Kullback. Information theory and statistics. Courier Corporation, (1997).
- [19] Blei D M, Ng A Y, Jordan M I. Latent dirichlet allocation. the Journal of machine Learning research, 2003(3): 993-1022.

How Instructional Feedback Has Been Employed in Instructional Units for Teaching Software Project Management Tools

A Systematic Literature Review

Rafael Queiroz Gonçalves

Department of Informatics and Statistics, Graduate Program on
Computer Science
Federal University of Santa Catarina, UFSC
Florianópolis, Brazil
rafael.queiroz@posgrad.ufsc.br

Christiane Gresse von Wangenheim

Department of Informatics and Statistics, Graduate Program on
Computer Science
Federal University of Santa Catarina, UFSC
Florianópolis, Brazil
c.wangenheim@ufsc.br

Abstract— The software industry has a growing demand for project managers which has leveraged the efforts to improve the teaching of project management competencies in higher education computer courses. Part of these efforts has been concentrated on improving instructional feedback when teaching these competencies. Instructional feedback is essential in order to help the students to learn based on an evaluation of their actions and decisions. However, so far there is few information available on how to provide this instructional feedback when teaching software project management. Therefore, we performed a systematic literature review that aims at providing an overview on existing feedback strategies, specifically when teaching the use of project management tools, such as MS Project or dotProject. As result we identified 8 relevant studies. Their results are systematically presented and a discussion is carried out, identifying the most adopted feedback strategies, how they were implemented, as well as their effects on student learning. These results may be useful for software engineering instructors to decide on the adoption some of the strategies as well as for instructional designers that may employ some of these feedback strategies when developing new instructional units.

Keywords—Project Management; PMBOK; PM tool; Instructional feedback; Instructional Unit; SLR.

I. INTRODUCTION

Software Project Management (PM) is emerging as a critical Software Engineering area for many software organizations. Many projects still fail due to a lack of proper management, leading to deadline or budget overrun, or the lack of scope coverage [1]. Therefore an essential requirement is to establish systematic PM using knowledge, abilities, tools and techniques that enable a project to reach its goals [1, 2, 3]. The adoption of a PM process may be aided by the usage of a PM tool [4]. A PM tool (e.g. MS Project or dotProject) is a software that either supports the whole PM process or any part of it. Typical functionalities include: schedule development, cost planning, risk analysis, etc. [9]. And, although, small software organizations often do not employ any PM tool, their potential benefits have increased the interest in their usage [5]. A possible cause for the shortfall of a broad adoption may be the lack of PM competencies of current project managers and

team members as they are often allocated without sufficient training [1, 4, 6].

However, PM knowledge and skills are an important competency of Software Engineering (SE) professionals [8], requiring the teaching of PM tools amongst the competencies defined in the ACM/IEEE reference curriculum for computer science [7]. In this context, few Instructional Units (IUs) aiming at the teaching of PM tools in higher education computer courses have been presented [10, 11]. An IU is a set of classes and instructional materials designed to teach certain learning objectives to a specific target audience [12]. Despite the progress, there are still gaps in this area, such as a limited content coverage and lack of students' motivation [13]. In this context, an important approach to improve students learning is the instructional feedback, which provides a response to student actions, indicating how they may improve it in order to achieve the IU's learning goals [14].

Aiming at the identification on how instructional feedback has been employed for teaching of PM tools, this study presents a Systematic Literature Review (SLR) [15], identifying existing IUs and analyzing and discussing their variations and benefits.

II. BACKGROUND

A. Project Management

PM directs project activities and its resources in order to meet the project requirements. It is organized in 5 process groups, from initiation to closure (Fig. 1) based on the PMBOK [2], as the main reference widely accepted worldwide [4].

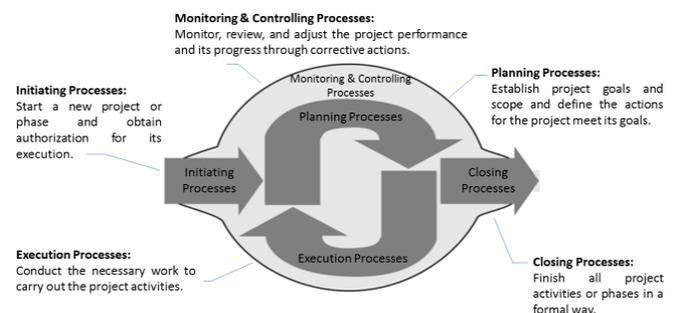


Figure 1. PM processes groups [2].

Orthogonally to the processes groups, PM processes are organized in 10 knowledge areas (TABLE I).

TABLE I. PM knowledge areas [2].

Knowledge area	Processes to:
Integration	Identify and coordinate PM processes and PM activities.
Scope	Ensure the project addresses the entire work to meet its requirements.
Time	Plan and control the activities that will be carried out during the project, so it concludes within the deadline.
Cost	Plan, estimate, and control project costs, so it concludes within the approved budget.
Quality	Define the goals and quality policies, so the project meets the needs that have initiated it.
HR	Organize and manage the project team.
Communication	Ensure the generation, collection and distribution of project information.
Risk	Identify and control the project risks.
Acquisition	Buy or contract products, services or any resources that are not available as project internal resources.
Stakeholder	Identify and manage the stakeholders and its expectations.

The application of a PM process is aided by the usage of PM tools, which take advantage from technology. Currently, many PM tools have been developed, such as MS-Project (microsoft.com/project), GanttProject (ganttproject.biz), dotProject (dotproject.net), Project.net (project.net), etc. [16]. The support provided by these tools may semi-automatize a few activities of the PM process such as registering and sequencing project activities, providing online forms to record their duration estimates or resource allocations, as well as generating a Gantt chart [5]. Some PM process activities may even be totally automated by PM tools, for instance, the calculation of the project total cost, the identification of the critical path or over-allocated resources [4, 10].

B. Teaching of Project Management

Knowing how to use PM tools is an important competency for project manager and SE professionals [2, 7, 8]. This includes the competency on how to use PM tools in order to develop a project schedule, to perform risk analysis, to monitor the project performance, etc. Often the teaching of PM tools also covers the following techniques [2, 6, 11]:

- **Critical Path Method (CPM):** to identify project activities that cannot be delayed without affecting the deadline;
- **Program Evaluation and Review Technique (PERT):** to calculate the estimated effort to carry out an activity based on three estimates (worst case, most common case, and best case);
- **Earned Value Analysis (EVA):** to measure project performance and progress in an objective manner;
- **Resources Leveling:** to adjust start and finish dates based on resource constraints, with the goal of balancing resource demand and availability.

Several approaches already have been developed to assist in the teaching of these competencies, such as educational PM tools. It may provide exercises (scenarios) that propitiate the carrying out of these PM techniques, the configuration of difficulty levels, and the adoption of instructional feedback, which may assist students during PM tools usage [6, 10, 11].

C. Instructional Feedback

Instructional feedback aims at assisting and stimulating students to reflect about their responses, providing information

to direct the student way of thinking and acting, thus promoting their learning [17].

There are different ways to promote feedback depending on the adopted instructional strategy [18]. A feedback may be verifying (informing if the student response is correct or incorrect) or elaborating (providing information to assist in the improvement of student response) [19]. Feedback may focus on different aspects of student response correctness, as it may be positive (highlight the students correct responses), negative (highlight the students mistakes) or constructive (not only informing if a certain response is wrong or incomplete, but also providing means for the student to improve it) [20].

Feedback may also have different goals. Formative feedback aims at instructing students during some activity, assisting in the response improvement before its delivery. On the other hand, summative feedback is given at the end of an activity, and is normally used for evaluation/assignments [21].

Feedback also may be classified according to the moment of its delivery to the student. Basically it can be immediate, when delivered just after some student response, or delayed, when provided some time after the conclusion of some activity [22]. Yet, the moment of delivering feedback is quite related to the adopted channel for its delivery, which may vary from computer-based, orally or written by the instructor. Especially in computer-based environments, immediate (automated) feedback has been an interesting alternative. On the other hand, when done manually by the instructor, generally delayed feedback is more suitable [23].

III. DEFINITION OF THE SYSTEMATIC LITERATURE REVIEW

Aiming at the identification of studies that have employed some instructional feedback strategy for teaching PM tools, we conducted a Systematic Literature Review (SLR). A SLR aims at the identification, analysis and interpretation of existing studies and that are relevant to the research question [15]. We followed the process defined by [15], organized in 4 phases:

1. **SLR Definition:** defining the research goal, the research questions, data sources, keywords, and inclusion/exclusion criteria for studies selection.
2. **SLR Execution:** based on defined keywords, queries are executed in each data source, and the relevant results are selected.
3. **Data extraction and synthesis:** data is extracted from each relevant study, aiming at the collection of data that support the answering of the research questions.
4. **Analysis and discussion:** the research questions are answered based on the analysis of the collected data, and the results are discussed achieving the research goal.

A. Research Question

With the objective to identify and characterize existing IUs for teaching PM tools that adopt some instructional feedback strategy to improve student learning, the following research questions are defined:

- **RQ1.** What are the existing IUs to teach PM tools, employing some instructional feedback strategy?
- **RQ2.** What are their instructional feedbacks?
- **RQ3.** What software tools are adopted to assist the delivery of feedback?

- **RQ4.** How the effectiveness of the feedback strategy has been evaluated?

B. Inclusion/Exclusion Criteria

We focused on articles published between January 2006 and January 2016 (a 10 years range), written in English language and published in journals or conference proceedings, assuring a peer review process. Studies not related to teaching PM tools were excluded, such as studies focused on teaching programming, agile methodologies, like XP and SCRUM, or exclusively focusing on PM theory or certifications. Due to our focus on the software domain, we also excluded any studies not related to computing.

C. Data Sources and Keywords

For our search we chose prominent data sources based on their relevance for SE and their availability via Portal CAPES – a web library provided by the Brazilian government, giving access to relevant international scientific production. Thus, the selected data sources are: IEEEExplore (ieeexplore.ieee.org), ACM Digital Library (dl.acm.org), ScienceDirect (sciencedirect.com), Wiley online library (onlinelibrary.wiley.com), SpringerLink (springerlink.com), and Scopus (scopus.com).

For the identification of relevant studies the following keywords and synonyms were defined based on the main concepts of the research goal (TABLE II).

TABLE II. Search keywords.

Concept	Keyword and synonymous
Feedback	Feedback, formative evaluation, formative assessment.
Education	Education, teaching, learning.
Project Management	Project management, PMBOK.
Tool	Tool, system, software.

IV. EXECUTION OF THE SLR

The defined SLR was carried out in January 2016 by the authors, and the results from each data source are presented in TABLE III.

TABLE III. Number of retrieval results per data source.

Data source	Number of retrieval results
ACM Digital Library	18
IEEEExplore	39
ScienceDirect	3
Scopus	26
SpringerLink	178
Wiley online library	15
Total	279

TABLE V. Characterization of Instructional Units (RQ1).

Study	Learning goal After the IU classes the students are able to:	Instructional activities
S1	Understand the trade-off between time, cost, scope, and its effects during the management of a project.	Students use a PM tool simulator to input information related to the project plan. Then, they run several simulation rounds and are presented with the effects of their planning to the project performance.
S2	Interpret the project performance indicators and take proper decisions based on these data.	Students use a PM tool simulator that presents information related to the project EVA. Then, the simulator includes a chatter bot that play the role of different stakeholders, such as team member, client, and sponsor, to whom the student has to provide the proper information based on the EVA information.
S3	Understand how to use a PM tool to register the project documents and use communication channels to communicate with other stakeholders.	Students organized in groups have to develop a software project and use a PM tool to register its documentation. The teamwork interactions (leadership, communication, feedback seeker, feedback provider) is automatically analyzed, resulting in an immediate feedback about the teamwork communication, based on a graphic dashboard.
S4	Understand how to use a PM tool to register the rationales about the PM decisions they have to take during the project planning and execution.	Students organized in groups have to develop a software project, planning it in a collaborative PM tool. Then, during its planning and execution, the rationales about all decision-making should be documented in this tool. The explanations provided by a student may be improved by others, leading to a collaborative feedback among students.
S5	Understand how to use a PM tool to evaluate the CPM and perform resource levelling technique.	Students use a PM tool simulator to run several simulation rounds and, between each round, students have to take decisions based on analysis of the CPM, Gantt chart, and resource leveling.
S6	Understand the whole PM process and apply the	Students have to carry out real software project derived from a university community service program. Organized in

These initial retrieval results were analyzed by their title and abstract and potentially relevant results were selected with respect to the defined inclusion/exclusion criteria. These results were analyzed in detail based on their full text, and again based on the inclusion/exclusion criteria. Many retrieval results were excluded because they do not focus on teaching PM tools but programming, agile methods, among others. Other results were excluded as they dealt with an evaluation of the UI interpreting the term “feedback” as the students’ opinion about some teaching strategy, and, thus, not related to the employment of some feedback strategy by an IU. Some studies rather focused on the employment of feedback techniques by project managers when managing the team members. Another significant amount of studies was excluded as they were not related to the computing area. As a result, we identified 8 relevant studies as presented in TABLE IV.

TABLE IV. Selected studies.

ID	Reference
S1	D. Rodriguez et al. E-Learning in Project Management Using Simulation Models: A Case Study Based on the Replication of an Experiment. IEEE Transactions on Education, vol. 49, n. 4, pp. 451-463, 2006.
S2	Y. Tachikawa et al. A method for evaluating project management competency acquired from role-play training. In: Proc. of IEEE Global Engineering Education Conference, Berlin/Germany, 2013.
S3	R. Vivian et al. The Development of a Dashboard Tool for Visualising Online Teamwork Discussions. In: Proc. of 37th IEEE International Conference on Software Engineering (ICSE), Florence/Italy, 2015.
S4	L. Xiao et al. Promoting Reflective Thinking in Collaborative Learning Activities. In: Proc. of 8th Int. Conf. on Advanced Learning Technologies, Cantabria/Spain, 2008.
S5	A. Chua, and R. Balkunje. An Exploratory Study of Game-based M-learning for Software Project Management Journal of Universal Computer Science, vol. 18, n. 14, pp. 1933-1949, 2012.
S6	G. Dixon. Service learning and integrated, collaborative project management. Project Management Journal, vol. 42, n. 1, pp. 42-58, 2011.
S7	G. Gregoriou, K. Kirytopoulos, and C.Kiriklidis. Project Management Educational Software (ProMES). Computer Applications in Engineering Education, vol. 21, n. 1, pp. 46-59, 2010.
S8	U. Ojiako et al. The criticality of transferable skills development and virtual learning environments used in the teaching of project management. Project Management Journal, vol. 42, n. 4, pp. 76-86, 2011.

V. DATA EXTRACTION AND SYNTHESIS

In alignment with the defined research questions we extracted the relevant information from the selected studies. With respect to the first research question we identified their learning goals and instructional activities, than characterizing the IUs (TABLE V).

	concepts in a practical software project.	groups they work as a project team and must plan and execute the project, receiving constant feedback from teachers.
S7	Understand how to apply CPM, PERT and RACI matrix techniques in a PM tool.	Students have to carry out exercises using a PM tool, which are related to CPM, PERT and RACI matrix techniques. The tool automatically evaluates student interactions and provides an automated feedback.
S8	Understand how to use a PM tool to define project scope, develop its schedule and HRs allocation.	Students use a PM tool simulator to deal with 5 different scenarios, each one containing some problem related to scope, time or HR knowledge areas. The IU also includes some exercise to be performed directly using MS-Project.

Once we identified the IUs, the adopted feedback strategies are characterized (TABLE VI). The information we extracted includes the feedback goal, its approach, the correctness evaluation, the moment and its delivery channel.

TABLE VI. Characterization of feedback strategies (RQ2).

Study	Goal	Approach	Correctness evaluation	Moment	Delivered Channel
S1	Formative	Elaborated	Constructive	Immediate	Computer-based
S2	Formative	Elaborated	Constructive	Delayed and Immediate	Computer-based and written by teacher
S3	Summative	Elaborated	Positive and negative	Immediate	Computer-based
S4	Formative	Verified	Constructive	Delayed	Computer-based and by other students
S5	Formative and summative	Elaborated	Constructive	Immediate and Delayed	Computer-based
S6	Formative and summative	Elaborated	Constructive	Delayed	Personally by teacher (orally and written)
S7	Formative	Verified	Negative	Immediate	Computer-based
S8	Summative	Verified	Positive and Negative	Immediate	Computer-based

Most of the identified IUs adopt some software tool for providing the instructional feedback. Therefore, we also have extracted data to characterize these tools (TABLE VII), including its main functionalities, educational features, etc.

TABLE VII. Characterization of educational software tools adopted to assist with feedback delivery (RQ3).

Study	Educational tool	Main PM functionalities	Educational features	Availability	Screen example
S1	Project simulator	Plan scope size, schedule, and HR allocation.	Runs simulations based on a project plan and students may analyze the simulations results to re-plan the project, aiming at conclude it successfully. The activities carried out in the tool are scenario driven, propitiating the application of PM techniques.	Not available	
S2	ProMASTER	Monitor a project using EVA.	Presents different stakeholders behavior by its chatterbot. And during the activities, it presents hints about calculation of EVA indicators.	Not available	Not presented
S3	PIAZZA	Manage communication by a document repository (wiki) and forums.	Although not being an educational tool, it has been adopted for educational proposes. Students use it to register project documentation, using wiki and forum structures. Later, this data is automatically extracted and analyzed, providing feedback to students.	Proprietary	
S4	Workspace	Register project communication.	Provides a shared whiteboard for students to register their decision rationales. Each rationale is discussed with other students that provide feedback to enhance the rationale ideas.	Not available	
S5	MAPLE	CPM, Gantt Chart, PERT, HR leveling.	Presents student performance based on remaining time and budget in the simulated software project. Periodically (each quarter of project duration) provides feedback to students. It also presents a glossary with the activity related vocabulary.	Not available	
S6	Does not adopt any software tool for feedback delivering.				
S7	ProMES	CPM, PERT, and RACI matrix.	Provides scenarios (exercises) for the application of PM techniques. Feedback is based on student interaction with the PM tool, indicating when they wrongly perform its functionalities, or take decisions which may lead to undesired results.	Open-source	
S8	HBSP	Plan scope size, schedule, and HR allocation.	Runs simulations based on a project plan, providing a dashboard for project performance tracking. Students may analyze this dashboard and re-plan the project aiming at conclude it successfully. The dashboard is a graphical representation of feedback related to students' performance.	Proprietary	

In order to aid the decision on the adoption of such feedback strategy, we also analyze the question on their impacts on the students' learning and, therefore, extracting data on how they were evaluated. This data is presented in TABLE VIII, describing the adopted evaluation method, how the data were collected, what were the analysis results. Among the evaluation methods are: *ad-hoc* (informally analyzing students reaction); case study (explicitly defining the evaluation and collecting data without a control group); and experiment (explicitly defining the evaluation and collecting data with a control group adopting another IU) [24]. Nonetheless, the evaluated dimensions (i.e. what was analyzed) were identified, including students learning and motivation, or their perception about instructional strategy effectiveness.

TABLE VIII. Data related to the feedback strategies evaluation RQ4.

Study	Evaluation goal	Evaluation method	Sample size	Data collection instrument	Evaluated dimensions	Evaluation results
S1	Evaluate students have learned with the assistance of the e-learning tool, with support for PM simulation and immediate feedback delivery.	Experiment	11	Pre-test and post-test: containing questions related to planning of software project.	<ul style="list-style-type: none"> Learning Motivation 	The students using the e-learning tool with the incorporated PM simulation model gained a better understanding about typical behavior patterns of software development projects.
S2	Evaluate student learning through the execution of role playing training.	Case study	27	The messages students have exchanged with the charter bot.	<ul style="list-style-type: none"> Learning Strategy Materials 	Student learning was evaluated according to a defined rubric. Feedback was perceived to make an important contribution on students learning.
S3	Evaluate student communication in a software project, through the analysis of documentation shared in a PM tool.	Case Study	26	The exchanged messages on the PM tool.	<ul style="list-style-type: none"> Learning Motivation 	The immediate feedback provided through the dashboard allows the students to spend less time analyzing the teamwork and more time exploring

						teamwork activity and team progression.
S4	Evaluate the students' capability to document their rationales using the communication channels of a PM tool.	Ad-hoc	30	Questionnaire (Likert scale)	<ul style="list-style-type: none"> • Learning • Strategy • Motivation 	The results show the students identified that the process of generating and sharing their rationales helped the group members to clarify their ideas, exploring the basic concepts of PM.
S5	Evaluate if the students have learned and got motivated when using MAPPLE tool.	Case Study	55	Questionnaire (closed and open-ended questions)	<ul style="list-style-type: none"> • Learning • Motivation • Strategy 	The feedback features provided by MAPPLE has contributed to student learning and the gaming aspects have contributed for student's motivation.
S6	Evaluate the impact of student participation in a real software project in their PM learning.	Case Study	-	Questionnaire (Likert scale)	<ul style="list-style-type: none"> • Learning • Strategy 	The students highlighted the feedback assisted them to be constantly engaged in the whole PM process during the project they were involved.
S7	Evaluate if students learned about the application of CPM PERT and RACI matrix techniques in PM tools.	Ad-hoc	20	Observation, Students testimony.	<ul style="list-style-type: none"> • Learning • Motivation • Strategy 	Students found the feedback very helpful. They stated the fact it indicated the problems of each solution, instead of revealing it, giving to them the opportunity to find the right answer.
S8	Evaluate student perception about their learning when carrying out the simulation exercises.	Case study	113	Questionnaire Likert scale	<ul style="list-style-type: none"> • Learning • Motivation 	Students considered the simulation a low-risk way for obtaining experience at managing projects. The formative feedback served to improve their learning.

VI. ANALYSIS AND DISCUSSION

In this section we analyze each research question in order to achieve the research goal defined.

RQ1. What are the existing IUs to teach PM tools, employing some instructional feedback strategy?

In general the learning goals of the IUs are related to the teaching of techniques related to scope, time, communication, and HRs knowledge areas (S1, S5, S7, S8), including CPM, PERT, EVA, etc. It is important to highlight that some IUs (S2, S3, S4) are centered on the communication knowledge area, providing communication channels among students, who may assume different stakeholder roles in a software project. All identified studies reported that they adopt an experiential learning instructional strategy, so students learn about PM tools using it, either by developing a software project (S3, S4, S6), or by simulating its execution (S1, S2, S5, S7, S8).

RQ2. What are their instructional feedbacks?

The instructional feedback is a response message to some student action. When analyzing how it is employed in the identified IUs, we observed that both the elaborating and verifying approaches have been adopted. It is evident that the elaborated feedback is more adopted, with few studies employing verified feedback, normally highlighting the errors in student responses (S4, S7, S8). In relation to correctness evaluation, constructive feedback is the most employed, with few studies employing the negative feedback, just highlighting students' errors, without indicating how to make improvements (S3, S7). Most studies provide immediate feedback, delivered by a software tool. In two studies delayed feedback was delivered by a teacher after the students finished some activity (S2, S6), or by the educational tool at specific milestones reached by students (S4, S5). Hence, we can observe that formative feedback is the most adopted, presenting explanations about the content, or pointing out the errors in student responses, before the conclusion of the activities. On the other hand, summative feedback was used to evaluate the students' performance at the end of activities, but normally without assigning marks (S3, S5, S8), just presenting project performance indicators such as the remaining budget, time or scope.

RQ3. What software tools are adopted to assist the delivery of feedback?

Most IUs have adopted some software tool for feedback delivery. These tools usually presented feedback in form of text boxes (for explanations) or graphical panels (for performance tracking). Here, we can also differentiate 2 main types of tools:

- PM tool simulators** (S1, S5, S7, S8): in which students may plan an project and simulate its execution, having the opportunity to re-plan the project after each execution round;
- Collaborative PM tools** (S2, S3, S4): in which students are organized in groups documenting their decisions, providing feedback to each other and in this way building the knowledge together.

Only S7 does not present the adoption of any educational tool for feedback delivery, providing it directly by the teacher. The main particularity of this study are the real software projects the students have to manage, due to a cooperation between the educational institution and software industry, which may have reduced the need for an educational tool.

RQ4. How the effectiveness of the feedback strategy has been evaluated?

In general the evaluations aimed at analyzing the effectiveness of instructional feedback in students learning. Moreover, many studies also analysed the students motivation and their perception about the instructional strategy. In relation to the evaluation method, most studies adopted the case study approach, where after have participated from the IU, the students were invited to answer a questionnaire, generally based on Likert scale items (S4, S6, S8).

Only study S1 performed an experiment, using the experiment group the proposed e-learning tool, and the control group the classic COCOMO model [25]. The students learning and motivation were evaluated, based on pre-test and post-test analysis. As a result, the students from the experimental group demonstrated a better learning than the control group, and were more motivated, presenting greater interest in the IU subject.

Thus, in general, we observed that although encountering some studies, there is still a lack of more comprehensive support of instructional feedback via an education PM tool.

As it seems that instructional feedback is welcome by the students and perceived by them to improve their learning, it certainly is an aspect worth to be explored more in order to make the learning process more effective. When being

automated as part of an educational tool it further more has the potential to reduce the instructor's effort. However, despite many advances already been implemented in this respect, there are still several improvement opportunities, especially in relation to content coverage, as most educational PM tools do not cover the whole PM process, including all processes groups and knowledge areas.

A. Threats to Validity

A common threat in any SLR is the bias inherent to scientific publications, due to the fact that most studies rather report successes, than failures. This threat may have hampered the identification of ways to evaluate the feedback effectiveness.

During the SLR search the main threat is to not find relevant studies. We tried to minimize this threat by including several data sources and the use of synonyms for all search keywords.

With respect to the selection of the studies, a principal threat is related to the influence of the researchers' personal opinion. We, therefore, explicitly defined inclusion/exclusion criteria beforehand. Furthermore the selection has been done by both authors in cooperation. This threat may impact data extraction, as some information is not explicit described in the articles, and had to be inferred by authors.

VII. CONCLUSION

This paper presents and overview on feedback strategies in IUs for teaching PM tools. As results of the systematic literature review, after analyzing more the 250 retrieval results, 8 relevant studies have been encountered. As a result, we observed that in general, instructional feedback is immediate and formative, providing the students hints or concept explanations to assist their understanding about the theory related to the instructional activities. Some studies demonstrated the usage of summative feedback, being graphically displayed at the end of activities. Based on their evaluations the studies also indicate that instructional feedback is perceived as an important tool to improve student learning and motivation. And, although there seem to exist only few research on this topic, future studies may take advantage of reported feedback strategies in the development of other IUs for teaching PM tools, thus improving the quality of the teaching of this important competency for SE professionals.

ACKNOWLEDGMENT

This work was supported by the CNPq (*Conselho Nacional de Desenvolvimento Científico e Tecnológico* – www.cnpq.br), an entity of the Brazilian government focused on scientific and technological development.

REFERENCES

[1] The Standish Group. Chaos Manifesto 2013, Boston, 2013.
 [2] PMI – Project Management Institute. A Guide to the Project Management Body of Knowledge, 5. ed., Newtown Square, 2013.
 [3] M. Keil, A. Rai, and J. Mann. Why software projects escalate: The importance of project management constructs. *IEEE Transactions on Engineering Management*, vol. 50, n.3, pp. 251–261, 2003.

[4] R. Fabac, D. Radošević, and I. Pihir. Frequency of use and importance of software tools in project management practice in Croatia. In: *Proc. of 32nd Int. Conf. on Information Technology Interfaces, Cavtat*, 2010.
 [5] H. Cıciabas, O. Unal, and K. Demir. A comparison of project management software tools (PMST). In: *Proc. of the 9th Software Engineering Research and Practice, Las Vegas*, 2010.
 [6] K. Reid, and G. Wilson. DrProject: A Software Project Management Portal to Meet Educational Needs. In: *Proc. of the Special Interest Group on Computer Science Education, Covington*, 2007.
 [7] ACM, IEEE Computer Society. *Computer Science Curricula 2013*, 2013
 [8] IEEE CS. *SWEBOK - Guide to the Software Engineering Body of Knowledge*. Los Alamitos, California, USA: IEEE, 2004.
 [9] Ž. Car, H. Belani, and Pripuzić K.. Teaching Project Management in Academic ICT Environments. In: *Proc. of the Int. Conf. on computer as a tool, Warsaw*, 2007.
 [10] L. Salas-Morera, A. Arauzo-Azofra, and L. García-Hernández. PpcProject: An educational tool for software project management. *Computers & Education*, vol. 69, n. 1, pp. 181-188, 2013.
 [11] G. Gregoriou, K. Kirytopoulos, and C. Kiriklidis. Project Management Educational Software (ProMES). *Computer Applications in Engineering Education*, vol. 21, n. 1, pp. 46–59, 2010
 [12] H. Hill, B. Rowan, and D. Ball. Effects of Teachers' Mathematical Knowledge for Teaching on Student Achievement. *American Educational Research Journal*, vol. 42, n. 2, p. 371-406, 2005.
 [13] R. Gonçalves, and C. Wangenheim. How to Teach the Usage of Project Management Tools in Computer Courses: A Systematic Literature Review. In: *Proc. of the Int. Conf. on Software Engineering and Knowledge Engineering, Pittsburgh*, 2015.
 [14] F. Kleij, T. Eggen, C. Timmers, and B. Veldkamp. Effects of feedback in a computer-based assessment for learning. *Computers & Education*, vol., n. 1, pp. 263–272, 2012.
 [15] B. Kitchenham, "Systematic literature reviews in software engineering – A tertiary study," *Information and Software Technology*, vol. 52, n. 1, pp. 792-805, 2010.
 [16] A. Mishra, and D. Mishra. Software Project Management Tools: A Brief Comparative View, *ACM SIGSOFT Software Engineering Notes*, 38(3), pp. 1-4, 2013.
 [17] M. Bilal, P. Chan, F. Meddings, and A. Konstadopoulou. SCORE: An advanced assessment and feedback framework with a universal marking scheme in higher education. In: *Proc. of the Int. Conf. on Education and e-Learning Innovations*, (pp. 1-6). Sousse, 2012.
 [18] R. Branch. *Instructional Design: The ADDIE Approach*. Springer, 2nd edition, 2009.
 [19] M. Segers, D. Gijbels, and M. Thurlings. The relationship between students' perceptions of portfolio assessment practice and their approaches to learning. *Educational Studies*, vol. 34, n. 1, pp. 35–44, 2008.
 [20] R. Martens, C. Brabander, J. Rozendaal, M. Boekaerts, and R. Leeden. Inducing mind sets in self-regulated learning with motivational. *Educational Studies*, vol. 36, n. 3, pp. 311–327, 2010.
 [21] A. Kretlow, and C. Bartholomew. Using coaching to improve the fidelity of evidence-based practices: A review of studies. *Teacher Education and Special Education*, vol. 33, n. 4, pp. 279–299, 2010.
 [22] M. Scheeler, M. Congdon, and S. Stansbery. Providing immediate feedback to co-teachers through bug-in-ear technology: An effective method for peer coaching in inclusion classrooms. *Teacher Education and Special Education*, vol. 33, n. 1, pp. 83-96, 2010.
 [23] Z. Fund. Effects of communities of reflecting peers on student–teacher development – Including in-depth case studies. *Teachers and Teaching: Theory and Practice*, vol.16, n. 6, pp. 679–701, 2010.
 [24] D. Tofan, Software engineering researchers' attitudes on case studies and experiments: An exploratory survey. In: *Proc. of 15th Annual Conference on Evaluation & Assessment in Software Engineering, Durham/England*, 2011.
 [25] B. Boehm, *Software Engineering Economics*. Englewood Cliffs, NJ: Prentice-Hall, 1981.

A Ranking-Oriented Approach to Cross-Project Software Defect Prediction: An Empirical Study

Guoan You and Yutao Ma*

State Key Laboratory of Software Engineering, Wuhan University, Wuhan 430072, China
School of Computer, Wuhan University, Wuhan 430072, China

*E-mail: ytma@whu.edu.cn

Abstract—In recent years, cross-project defect prediction (CPDP) has become very popular in the field of software defect prediction. It was treated as a binary classification or regression problem in most of previous studies. However, the existing methods to solve this problem may be not suitable for those projects with limited manpower and time. In this paper, we revisit the issue and treat it as a ranking problem. Inspired by the idea of the Point-wise approach to Learning to Rank, we propose a ranking-oriented CPDP approach called ROCPDP. The empirical results obtained based on AEEEM show that the defect predictor built with our method under a specific CPDP context, in general, outperforms those predictors trained by using the benchmark methods in both CPDP and WPDP (within-project defect prediction) scenarios in terms of two common evaluation metrics for rank correlation. So, our work could be an initial attempt to construct new ranking-oriented CPDP models for newly created or inactive projects.

Keywords—cross-project defect prediction; ranking; single-objective optimization; gradient descent; multiple linear regression

I. INTRODUCTION

Due to the importance of software defect prediction in quality assurance and software maintenance, in recent years it has gradually become one of the most active research fields in software engineering [1]. To the best of our knowledge, there are two mainstream techniques for software defect prediction, namely, within-project defect prediction (WPDP) and cross-project defect prediction (CPDP).

Generally speaking, WPDP has one major disadvantage that the training of WPDP models needs sufficient historical data from the same project [2]. Therefore, it is difficult to apply WPDP models to those newly created or inactive projects. With the increasing amount of labeled defect data available on the Internet, till now, CPDP has already been the most popular technique for software defect prediction, though it is still getting criticized for relatively poor prediction performance compared with WPDP [3, 4].

The vast majority of prior studies on CPDP treat software defect prediction as a binary classification problem [1-8], and their main objective is to improve the prediction accuracy of CPDP models using various machine learning techniques [4, 6-8], such as feature selection, dimensionality reduction, and data sampling. However, estimating the defect-proneness of a given set of classes or software modules has limited effect on actual activities in software testing and software maintenance [7, 9, 10], especially when there is a lack of human resources. That is to say, from a software developer's point of view, a ranking list

of defect-prone software entities is definitely more useful than the information about how many the software entities in question are possibly buggy.

Motivation: Now take a typical application scenario as an example. Two software developers develop a new software project written in Java, and one of the developers (whose name is Jack) is responsible for software testing. Due to the tight deadline for a new release (composed of 1000+ Java classes), a sound technical solution for Jack is to identify the classes of the release that are most likely to be defect-prone before executing unit tests. Therefore, Jack builds a CPDP model based on commonly-used software metrics (such as lines of code and CK suite metrics [11]) using the Naïve Bayes (NB) classification algorithm. According to the prediction result of the CPDP model trained by other similar mature projects, only a very small percentage (about 7%) of the classes in question may be buggy, but the efficiency of the entire testing process is still relatively low because Jack has to perform random testing. In such a situation, Jack actually prefers to obtain a class ranking list that identifies the priority of each defect-prone class, so as to work out a cost-effective testing plan.

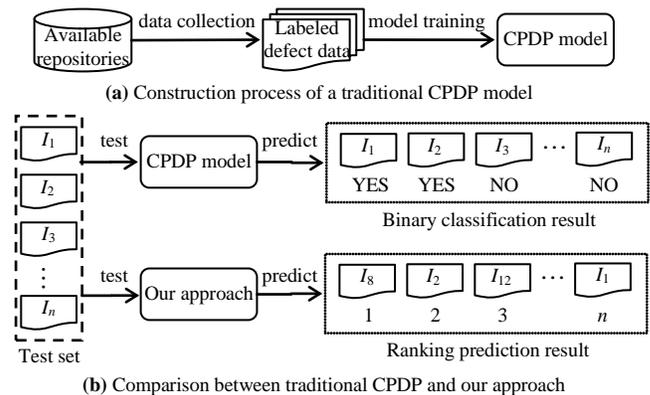


Figure 1. An illustration of the difference between various CPDP methods

Research Objective: Unlike the studies of CPDP based on binary classification, in this paper we focus on the prediction of ranks of defect-prone software entities. An illustration of the difference between them is shown in Figure 1. In short, the main goal of this paper is to propose a ranking-oriented approach to CPDP (abbreviated as ROCPDP), which can rank buggy software entities according to priority. In addition, we also want to validate the feasibility of ROCPDP using a case study based on five open-source projects from the publicly-available data set AEEEM [12].

The contributions of this paper are summarized below.

(1) We consider CPDP as a ranking (prediction) problem and propose a new, easy-to-use method called ROCPDP, which utilizes a simple multiple linear regression model with single-objective optimization. Besides, we optimize the model's parameters using the gradient descent method.

(2) We conduct an empirical study based on AEEEM, and the result shows that in different CPDP scenarios our approach outperforms other benchmark methods in terms of two common evaluation metrics, i.e., Spearman's rank correlation coefficient and Kendall rank correlation coefficient.

The rest of this paper is organized as follows: Section II introduces the related work of CPDP, and Section III presents the details of our approach; in Section IV, a case study is given to demonstrate that our approach is better than the benchmark methods with regards to the given evaluation metrics; potential threats to the validity of our study are presented in Section V; in the end, Section VI summarizes this paper and outlines future work directions.

II. RELATED WORK

As mentioned before, CPDP is a popular defect prediction technique used to predict defects in a given project with the prediction models (also called predictors) trained based on labeled defect data from other projects [13]. After Briand *et al.* made an early attempt to validate the applicability of CPDP [14], many researchers in this field have tried to improve the performance of CPDP models using different techniques such as data mining and machine learning. Fortunately, recent studies have shown that it is indeed a feasible method for defect prediction in software projects with different sizes [13, 15-20]. Due to space limitations of this paper, for more details about CPDP approaches, please refer to the latest surveys [6, 7].

In these CPDP scenarios, the focus of the researches most relevant to the topic of this paper is on predicting or estimating the number of software defects/faults in a given software entity, which could be deemed as a specific problem with predictive analytics [10, 21]. In fact, this is not a new field of study. Earlier studies often trained this type of predictors using linear regression [22] or multiple regression analysis [23]. That is to say, according to training examples, these predictors were used to estimate the relationships among variables, including a dependent variable, viz. the number of defects, and one or more independent variables, viz. software metrics. More precisely, such predictors help us predict the unknown value of dependent variable from the known values of independent variables.

Subsequently, many other types of predictors have been built with various regression methods such as decision trees (DT), support vector regression (SVR), and Bayesian ridge regression (BRR). Moreover, our prior empirical study [10] shows that in WPDP and CPDP scenarios the DTR-based predictor is the best estimator for the number of defects among the six prediction models under discussion. To further improve the accuracy of predictions, some optimization methods have been applied to the construction of these types of predictors. For example, Wang *et al.* [24] applied the Discrete Time Markov Chain (DTMC) model to predict the number of defects

at each state in future based on a defect state transition model; Rathore *et al.* [25] presented an approach to predicting the number of faults using Genetic Programming.

On the other hand, in recent years a few researchers tried to investigate software defect prediction from a new perspective. That is, they formulated this problem as a ranking problem rather than a binary classification problem or a regression problem [9, 21]. Therefore, the approaches to Learning to Rank (LTR) [26] in the field of information retrieval have been recently introduced to investigate software defection prediction. However, the above studies were conducted based on the assumption that the rank of a defect-prone software entity is proportional to the actual number of defects that it contains. In addition, they were also performed in WPDP scenarios using the conventional validation (e.g., partitioning the data set from a project into two sets of 80% for training and 20% for test) or 10-fold cross-validation. Hence, it is still unknown whether these methods proposed in prior studies [9, 21] can be used in practical CPDP scenarios.

III. A RANKING-ORIENTED CPDP APPROACH

Considering the research objective of this paper, we focus on the method for predicting buggy software entities' ranks in CPDP scenarios. Unlike previous similar studies [9, 21], we assume that the rank of a buggy software entity is proportional to the score determined by both the number of defects within the entity and the severity of each defect. The higher the score of a software entity receives, the higher its rank becomes.

A. Problem Definition

First of all, we build a predictor based on training examples from different software projects. Suppose that there are two ranking lists L_r and L_p for a given test set. L_r and L_p denote the actual and predicted results of defect-prone software entities, respectively, and they are partially ordered sets sorted in descending order according to the score of each software entity. So, the problem of this paper is how to minimize the difference between L_p and L_r . In other words, we want to make L_p approximate L_r as closely as possible, formally defined as

$$\max \text{sim}(L_p, L_r), \quad (1)$$

where sim is a function used to calculate the similarity between two given ranking lists.

B. Description of Our Approach

Inspired by the idea of the Point-wise approach to LTR [26], in this paper the above problem can be approximated by a regression problem: given the values of a set of software metrics for a software entity, predict its score. This implies that our approach is required to predict the scores of defect-prone software entities as accurately as possible. We then present the formal definition of actual/real scores as follows:

$$s_i^r = \begin{cases} \sum_{j=1}^k w_j, & \text{if the } i^{\text{th}} \text{ software entity has } k \text{ bugs,} \\ 0, & \text{if the } i^{\text{th}} \text{ software entity is non-defective.} \end{cases} \quad (2)$$

where w is the severity score of a bug and k is a positive integer. Note that w can be rated on a five-point or ten-point scale to score a bug's severity.

Because multiple linear regression (MLR) models have proven useful in software defect prediction [9, 27, 28], in this paper we define a simple MLR model predicting the scores of a given set of software entities, as described below.

$$s = \theta_0 + \theta_1 m_1 + \theta_2 m_2 + \dots + \theta_d m_d = \sum_{i=0}^d \theta_i m_i, \quad (3)$$

where m_i denotes the value of a given software metric, d indicates the number of software metrics, and θ_i is an unknown parameter of the model.

During the supervised learning process of a MLR model, the main goal of our method is to minimize all possible differences between values predicted by the model and the corresponding actual/real values. According to the principle of least squares, the estimation of model parameters in our method can be formulated as the minimization of a loss function, which is defined below.

$$L(\boldsymbol{\theta}) = \frac{1}{2} \left[\sum_{i=1}^N (s_i - s_i^r)^2 + \lambda \cdot \|\boldsymbol{\theta}\|^2 \right], \quad (4)$$

where $\boldsymbol{\theta}$ is a vector written as $(\theta_0, \theta_1, \theta_2, \dots, \theta_d)$, N is the number of training examples, and λ is a small regularization parameter.

C. Estimation of Model Parameters

As far as we know, gradient descent is a widely used first-order optimization algorithm. In this paper, we utilize it to find a local minimum of the above loss function, so as to estimate the unknown parameters of the model. For a given training set, the derivative with respect to θ of $L(\boldsymbol{\theta})$ is denoted as follows:

$$\nabla_{\boldsymbol{\theta}} L(\boldsymbol{\theta}) = \begin{bmatrix} \frac{\partial}{\partial \theta_0} L(\boldsymbol{\theta}) \\ \vdots \\ \frac{\partial}{\partial \theta_d} L(\boldsymbol{\theta}) \end{bmatrix}, \quad (5)$$

where the derivative with respect to any parameter θ_j of $L(\boldsymbol{\theta})$ is

$$\begin{aligned} \frac{\partial L(\boldsymbol{\theta})}{\partial \theta_j} &= \frac{\partial}{\partial \theta_j} \left[\frac{1}{2} \sum_{i=1}^N (s_i - s_i^r)^2 + \frac{\lambda}{2} \cdot \sum_{j=0}^d \theta_j^2 \right] \\ &= 2 \cdot \frac{1}{2} \sum_{i=1}^N \left[(s_i - s_i^r) \cdot \frac{\partial}{\partial \theta_j} (s_i - s_i^r) \right] + 2 \cdot \frac{\lambda}{2} \cdot \theta_j \\ &= \sum_{i=1}^N \left[(s_i - s_i^r) \cdot \frac{\partial}{\partial \theta_j} \left(\sum_{i=0}^d \theta_i m_i - s_i^r \right) \right] + \lambda \theta_j \\ &= m_j \cdot \sum_{i=1}^N (s_i - s_i^r) + \lambda \theta_j. \end{aligned} \quad (6)$$

Because the gradient descent method often takes multiple iterations to calculate a local minimum that meets the demand of accuracy, we update the value of θ_j in the negative direction of the gradient with a small learning rate (or known as step size) on each iteration, as defined below.

$$\theta_j := \theta_j - \alpha \cdot \frac{\partial L(\boldsymbol{\theta})}{\partial \theta_j} = \theta_j - \alpha \cdot \left[m_j \cdot \sum_{i=1}^N (s_i - s_i^r) + \lambda \theta_j \right], \quad (7)$$

where α is the learning rate ($\alpha > 0$) and the symbol ":= " denotes an assignment operator.

Algorithm 1: Estimation of Model Parameters Using Gradient Descent

Input: $\boldsymbol{\theta}$, α , λ , and N training examples with d metrics and a score
Output: $\boldsymbol{\theta}$

- 1: Initialize $\boldsymbol{\theta}$, α , and λ ;
 - 2: Randomly shuffle the training examples;
 - 3: **repeat**
 - 4: Calculate the value of s according to Eq. (3);
 - 5: Calculate the gradient of $\boldsymbol{\theta}$ according to Eq. (6);
 - 6: Update the value of $\boldsymbol{\theta}$ according to Eq. (7);
 - 7: **until** convergence
 - 8: return $\boldsymbol{\theta}$;
-

Figure 2. An algorithm to estimate model parameters

Figure 2 displays the algorithm used for estimating the model's parameters. Once all of the unknown parameters are fixed based on training examples, the entire learning process ends, and such a MLR model can be used for prediction. It is worth noting that evaluating the sum of gradients (see Eq. (6)) becomes very expensive if training examples are enormous. In this case, we can also utilize stochastic gradient descent to optimize the model's parameters when dealing with very large-scale data sets. That is, the gradient of θ_j is approximated by a gradient at a single example, thus leading to a low computation cost. Therefore, Eq. (6) can be rewritten as

$$\frac{\partial L(\boldsymbol{\theta})}{\partial \theta_j} \approx m_j \cdot (s_i - s_i^r) + \lambda \theta_j. \quad (8)$$

IV. CASE STUDY

A. Data Collection

The data set used in our case study is AEEEM [12], which consists of five open-source software projects written in Java and is available for free download on the Internet (website: <http://bug.inf.usi.ch>). Data statistics of the projects in this data set are presented in Table I. More details of the data set refer to the literature [4, 12].

TABLE I. DATA STATISTICS OF AEEEM

Project	Type	# of Files	% of Buggy Files	# of Metrics
Equinox (EQ)	OSGi framework	325	36.69%	71
Eclipse JDT Core (JDT)	IDE Development	997	20.66%	71
Eclipse PDE UI (PDE)	IDE Development	1492	14.01%	71
Mylyn (MYL)	Task management	1862	13.16%	71
Apache Lucene (AL)	Search engine library	399	9.26%	71

B. Experiment Design

Data preprocessing. There are six types of software metrics in AEEEM, including process metrics, entropy-of-source-code metrics, entropy-of-change metrics, churn-of-source-code metrics, previous-defect metrics, and source code metrics. Because they are different in the scales of numerical values, we have to normalize the raw data from AEEEM. Although there exist many normalization methods for machine learning and data mining, prior studies [10, 13, 16, 17, 18] have proven that the normalization of training data using z-score can, to some extent, improve prediction performance of defect classification models. Hence, in this paper we apply z-score normalization to every example in AEEEM.

Experiment Context Configuration. Unlike the prior studies that focus on WPDP [9, 21], in CPDP scenarios we design two types of experiment contexts, namely, one-to-one (O2O) and many-to-one (M2O). For a target project treated as a test set, the training set is a project chosen randomly from the rest of the projects in AEEEM, which is referred to as O2O. On the contrary, M2O indicates that the remaining AEEEM projects (except the target project) are used as the training set. Because some prior studies [2, 10, 13] have shown that defect prediction performs well within projects and cross projects when there is a sufficient amount of training data, in this paper we also want to examine whether our method can achieve better performance under the context of M2O.

Defect Predictor Training. According to training data, three types of defect predictors are trained in the experiment contexts. The first is simply built with a few typical regression methods such as DT and SVR, the second is trained by using two classic

optimization algorithms (i.e., gradient descent and genetic algorithm), and the third is obtained by using LTR approaches like RankSVM [29]. The detailed introduction to the above methods/algorithms refers to subsection IV.C.

Performance comparison. For each project in AEEEM, we can obtain four (C_{5-1}^1) prediction results and one prediction result under the contexts of O2O and M2O, respectively. Such prediction results are evaluated in terms of the metrics defined in subsection IV.D. To make a reasonable comparison between our approach and the benchmark methods, we will make twenty (5×4) O2O predictions and five (5×1) M2O predictions, respectively, and then analyze their mean values of the evaluation metrics for these predictions under different CPDP contexts (see Table III).

C. Learning Methods

In general, regression analysis methods are a category of quantitative methods which can predict the outcome of a given dependent variable according to the relationships with other related independent variables. In this paper, we build the first type of defect predictors with five typical regression analysis methods, namely, Linear Regression (LR), Random Forests (RF), Gradient Boosting (GB), DT, and SVR, and they are used to predict the score of each example in the test set. Due to space limitations of this paper, if readers are interested in the details of these methods, their descriptions refer to the corresponding Wikipedia entries. Note that all of the five methods are implemented with Weka [30]. Unless otherwise specified, the default parameter settings for these regression methods used in our experiments are specified by Weka.

TABLE II. BRIEF INTRODUCTION TO BENCHMARK APPROACHES

Approach	Brief Description
LR	Linear Regression is a widely used statistical approach to modeling the relationship between a scalar dependent variable y and one or more independent variables denoted as X , and it focuses on the conditional probability distribution of y given X . Moreover, the effectiveness of this method for predicting the number of bugs has been validated by prior studies [22, 23, 27, 28].
RF	Random Forests, known as a notion of the general technique of random decision forests, are an ensemble learning method for regression and other tasks. For regression analysis, random forests are used by building a number of decision trees according to training data and generating an outcome that is the mean prediction of the individual decision trees, and they have been utilized as a benchmark method in [9, 10, 27].
GB	Gradient Boosting, which is a well-known machine learning technique for regression and classification problems, combines weak prediction models into a single prediction model that has a greater capacity to deal with a given task, in a stage-wise fashion, and it often generalizes them by optimizing an arbitrary differentiable loss function. Moreover, it has been utilized as a benchmark method in [10].
DT	Decision trees are one of the common predictive modeling approaches used in data mining and machine learning, and they are of two main types: classification tree and regression tree. When predicted results are continuous values (typically real numbers), this type of decision trees is called regression tree. Decision trees have been used as a benchmark method in [9, 10, 21, 27].
SVR	Support vector machines (SVMs) are supervised learning models that can be used for classification and regression analysis, and a version of SVM for regression is called support vector regression (SVR). The models built based on SVR require only a small subset of training data and maintain all the main features that characterize the maximum-margin algorithm. Moreover, SVR has been used as a benchmark method in [10].
GA	An approach to predicting the number of bugs in software systems using Genetic Programming is presented in [25], and its effectiveness has been evaluated in terms of three evaluation metrics (namely, average relative error, Recall, and Completeness) for ten data sets collected from PROMISE (http://promisedata.org).
RankSVM	RankSVM is an instance of SVM for efficiently training Ranking SVMs as defined in [29], which is used to solve certain ranking problems. In general, it belongs to one of the Pair-wise ranking methods in information retrieval field. Moreover, RankSVM has been utilized as a benchmark method in [21].
RankBoost	RankBoost [31] is a boosting method for ranking a given set of examples. It trains one weak learner that produces a weak ranking at each iteration, and combines these weak rankings as the final ranking function. Like all boosting algorithms, RankBoost adjusts the weights assigned to pairs of examples after each round of iteration. Moreover, RankBoost has been utilized as a benchmark method in [21].

Obviously, the above defect predictors are learned without additional optimization. To improve the accuracy of CPDP predictors while maintaining their generality, we train the second type of defect predictors with typical single-objective

optimization algorithms. In addition to gradient descent used in this paper, there are actually many optimization algorithms for the single-objective optimization problem, such as Genetic Algorithm (GA) and Evolutionary Programming (EP). Since

GA is the most popular type of evolutionary algorithms, we optimize this problem with GA for the purpose of comparing with the prior study [25] similar to our work.

In a word, the ranks of test examples are calculated based on their scores which are predicted by the two types of defect predictors described above. Note that if two or more examples in test set get the same score, they will be sorted by example number in ascending order by default. In contrast, LTR-based defect predictors are obtained by optimizing the performance of ranking directly. In this paper, we utilize two commonly used Pair-wise approaches to LTR (i.e., RankSVM [29] and RankBoost [31]) in the field of information retrieval to train defect predictors.

The brief introduction to these benchmark approaches used in our experiments is presented in Table II.

D. Evaluation Metrics

Here, we choose two widely used statistical indicators to quantify the results of defect predictors. One is the Spearman’s rank correlation coefficient (*Spearman* for short), which is defined as the Pearson correlation coefficient between the ranked variables; the other is the Kendall rank correlation coefficient (*Kendall* for short), which measures the association between two measured quantities [32]. The value of each evaluation metric ranges between -1 and 1, and higher values closer to 1 mean better prediction performance.

E. Empirical Results

Because the data set AEEEM does not actually contain the information about the severity of bugs, we have to make the assumption that the bugs in AEEEM are all identical. That is

to say, the score of a class is equal to the number of bugs that it contains. Hence, all the buggy classes in AEEEM are ranked in terms of the number of bugs.

TABLE III. PREDICTION RESULTS UNDER CPDP CONTEXTS

	<i>Spearman</i>		<i>Kendall</i>	
	O2O	M2O	O2O	M2O
LR	0.261	0.287 (+9.96%)	0.217	0.220 (+1.38%)
RF	0.288	0.313 (+8.68%)	0.231	0.253 (+9.52%)
GB	0.225	0.239 (+6.22%)	0.180	0.181 (+0.56%)
DT	0.280	0.319 (+13.93%)	0.223	0.261 (+17.04%)
SVR	0.242	0.259 (+7.02%)	0.194	0.205 (+5.67%)
ROCPDP	0.416	0.449 (+7.93%)	0.357	0.392 (+9.80%)
GA	0.299	0.323 (+8.03%)	0.240	0.258 (+7.50%)
RankSVM	0.281	0.308 (+9.61%)	0.229	0.256 (+11.79%)
RankBoost	0.289	0.312 (+7.96%)	0.233	0.269 (+15.45%)

There are two interesting findings that deserve attention, as presented in Table III.

(1) For each of the nine approaches under discussion, the defect predictor trained under the context of M2O performs better than that trained under the context of O2O in terms of the two evaluation metrics. In particular, the defect predictor built based on DT achieves the most significant improvement of prediction performance under different CPDP contexts (as indicated by the numbers in bold in brackets). The finding, in agreement with previous studies on defect classification and regression [2, 10, 13], suggests that the prediction performance can be improved as long as there is sufficient training data collected from other projects.

(2) According to the two evaluation metrics, our approach, by and large, outperforms the other eight approaches under the two experiment contexts, as shown by the numbers in bold.

TABLE IV. PERFORMANCE COMPARISON AMONG DIFFERENT METHODS

	LR		RF		GB		DT		SVR		ROCPDP		GA		RankSVM		RankBoost	
	S	K	S	K	S	K	S	K	S	K	S	K	S	K	S	K	S	K
JDT	0.437	0.385	0.424	0.363	0.321	0.277	0.451	0.412	0.304	0.245	0.455	0.395	0.429	0.366	0.434	0.380	0.447	0.408
EQ	0.513	0.441	0.490	0.422	0.393	0.323	0.464	0.401	0.343	0.295	0.562	0.503	0.444	0.399	0.548	0.491	0.536	0.474
AL	0.222	0.167	0.267	0.223	0.198	0.134	0.246	0.193	0.257	0.218	0.391	0.355	0.356	0.304	0.211	0.162	0.232	0.173
MYL	0.252	0.198	0.196	0.137	0.189	0.132	0.278	0.212	0.230	0.183	0.486	0.413	0.304	0.247	0.237	0.185	0.249	0.198
PDE	0.291	0.226	0.275	0.211	0.231	0.182	0.319	0.231	0.238	0.179	0.351	0.294	0.364	0.295	0.321	0.271	0.308	0.269
Avg.	0.342	0.283	0.330	0.271	0.266	0.210	0.352	0.290	0.274	0.224	0.449	0.392	0.379	0.322	0.350	0.298	0.354	0.304

S: *Spearman*, K: *Kendall*, and the best result determined by both the two evaluation metrics is highlighted in bold type.

Because our prior study [10] has shown that CPDP models for bug numbers are comparable to (or sometimes better than) those WPDP models with respect to prediction performance. That is, there are actually no significantly statistical differences between the two types of defect predictors. To further validate the effectiveness of our method, we compared our method performed under the context of M2O with the benchmark approaches conducted in WPDP scenarios.

We trained eight new defect predictors in a specific WPDP scenario that has been used in [21]. For each target project in Table IV (see the first column), we randomly selected 80% of class files of the project as training data, and the remainder of the files was used as test data. We repeated this step 20 times and used the average of prediction results for the 20 executions as the final result. As shown in Table IV, because our method obtains the best result three times under the context of M2O, it

is, on average, better than the other eight approaches performed in the WPDP scenario in terms of both the two evaluation metrics. Compared with these benchmark methods, the performance improvements with respect to *Spearman* range from 18.47% (on GA) to 68.80% (on GB), and they span from 21.74% (on GA) to 86.67% (on GB) in terms of *Kendall*. Generally speaking, according to the results of Tables III and IV, under the context of M2O our method outstrips the benchmark methods conducted in both CPDP and WPDP scenarios, which suggests that it can be applied to those projects with very little historical defect data.

V. THREATS TO VALIDITY

The most interesting result of this paper is that the defect predictor built with our method under the context of M2O is, on average, better than those trained by using different types

of typical learning methods in both CPDP and WPDP scenarios. Even so, there are still some potential threats to the validity of our work, one of which concerns the generalization of the finding. The reasons mainly lie in the following aspects: 1) only the 5 projects in AEEEM is used in our experiments, implying that we need to validate the general effectiveness of our method on larger data sets from real-world software projects; 2) because the information about the severity of bugs is missing in AEEEM, we have to rank all the examples in question in terms of the number of bugs instead of its actual score; 3) in the WPDP and CPDP scenarios, we select training data in a simple way, though there are many time-consuming but effective methods for feature selection and dimension reduction [21]; and 4) we only utilize eight typical methods that belong to three different categories, namely, regression analysis, single-objective optimization, and learning to rank, without additional optimization for a given data set.

VI. CONCLUSION AND FUTURE WORK

In recent years CPDP has become very popular in software engineering, but it is often treated as a binary classification or regression problem in most of prior studies. To facilitate actual development activities in projects with limited manpower and time, in this paper we consider it as a ranking problem and propose a ranking-oriented CPDP method. Interestingly, the empirical results obtained based on AEEEM show that the defect predictor built with our method under the context of M2O is, on average, better than those trained by using the benchmark methods in both CPDP and WPDP scenarios. This suggests that our method is a sound choice to train defect predictors for those newly created or inactive software projects.

As an initial attempt to construct ranking-oriented defect predictors in different CPDP scenarios, the proposed approach in this paper needs to be improved. So, our future work is to accurately predict the Top- k ranking for buggy software entities.

ACKNOWLEDGEMENT

This work is supported by the 973 Program of China (No. 2014CB340404) and the National Natural Science Foundation of China (Nos. 61272111 and 61273216).

REFERENCES

- [1] C. Catal, "Software fault prediction: a literature review and current trends," *Expert Systems with Applications*, 2011, 38 (4): 4626–4636.
- [2] T. Zimmermann, N. Nagappan, H. Gall, E. Giger, and B. Murphy, "Cross-project defect prediction: a large scale experiment on data vs. domain vs. process," in: *ACM Proc. of ESEC/FSE 2009*, pp. 91–100.
- [3] T. Hall, S. Beecham, D. Bowes, D. Gray, and S. Counsell, "A systematic review of fault prediction performance in software engineering," *IEEE Transactions on Software Engineering*, 2012, 38 (6): 1276–1304.
- [4] M. D'Ambros, M. Lanza, and R. Robbes, "Evaluating defect prediction approaches: a benchmark and an extensive comparison," *Empirical Software Engineering*, 2012, 17(4-5): 531–577.
- [5] D. Radjenović, M. Heričko, R. Torkar, and A. Živković, "Software fault prediction metrics: A systematic literature review," *Information and Software Technology*, 2013, 55(8): 1397–1418.
- [6] G. Abaei and A. Selamat, "A survey on software fault detection based on different prediction approaches," *Vietnam Journal of Computer Science*, 2014, 1(2): 79–95.

- [7] R. Malhotra, "A systematic review of machine learning techniques for software fault prediction," *Applied Soft Computing*, 2015, 27: 504–518.
- [8] H.J. Wang, T.M. Khoshgoftaar, and Q.A. Liang, "A study of software metric selection techniques: stability analysis and defect prediction model performance," *International Journal on Artificial Intelligence Tools*, 2013, 22(5): 1360010.
- [9] X. Yang, K. Tang, and X. Yao, "A Learning-to-Rank Approach to Software Defect Prediction," *IEEE Transactions on Reliability*, 2015, 64(1): 234–246.
- [10] M.M. Chen and Y.T. Ma, "An empirical study on predicting defect numbers," in: *Proc. of SEKE 2015, USA*, 2015, pp. 397–402.
- [11] S.R. Chidamber and C.F. Kemerer, "A metrics suite for object oriented design," *IEEE Trans. on Software Engineering*, 1994, 20(6): 476–493.
- [12] M. D'Ambros, M. Lanza, and R. Robbes, "An extensive comparison of bug prediction approaches," in: *IEEE Proc. of MSR 2010*, pp. 31–41.
- [13] P. He, B. Li, X. Liu, J. Chen, and Y.T. Ma, "An empirical study on software defect prediction with a simplified metric set," *Information and Software Technology*, 2015, 59: 170–190.
- [14] L.C. Briand, W.L. Melo, and J. Wüst, "Assessing the applicability of fault-proneness models across object-oriented software projects," *IEEE Transactions on Software Engineering*, 2002, 28(7): 706–720.
- [15] J. Nam, S.J. Pany, and S. Kim, "Transfer Defect Learning," in: *IEEE/ACM Proc. of ICSE 2013, USA*, 2013, pp. 382–391.
- [16] B. Turhan, A.T. Misirli, and A. Bener, "Empirical evaluation of the effects of mixed project data on learning defect predictors," *Information and Software Technology*, 2013, 55(6): 1101–1118.
- [17] Z. He, F. Shu, Y. Yang, M.S. Li, and Q. Wang, "An investigation on the feasibility of cross-project defect prediction," *Automated Software Engineering*, 2012, 19(2): 167–199.
- [18] F. Peters, T. Menzies, and A. Marcus, "Better cross company defect prediction," in: *IEEE Proc. of MSR 2013, USA*, 2013, pp. 409–418.
- [19] S. Herbold, "Training data selection for cross-project defect prediction," in: *ACM Proc. of PROMISE 2013, USA*, 2013, p. 6.
- [20] F. Rahman, D. Posnett, and P. Devanbu, "Recalling the imprecision of cross-project defect prediction," in: *ACM Proc. of FSE 2012*, p. 61.
- [21] T.T. Nguyen, T.Q. An, V.T. Hai, and T.M. Phuong, "Similarity-based and rank-based defect prediction," in: *IEEE Proc. of ATC 2014*, pp. 321–325.
- [22] J.E. Gaffney Jr., "Estimating the Number of Faults in Code," *IEEE Transactions on Software Engineering*, 1984, 10(4): 459–465.
- [23] N.E. Fenton and M. Neil, "A Critique of Software Defect Prediction Models," *IEEE Trans. on Software Engineering*, 1999, 25(5): 675–689.
- [24] J. Wang and H. Zhang, "Predicting defect numbers based on defect state transition models," in: *ACM Proc. of ESEM 2012*, pp. 191–200.
- [25] S.S. Rathore and S. Kumar, "Predicting Number of Faults in Software System using Genetic Programming," *Procedia Computer Science*, 2015, 62: 303–311.
- [26] T.-Y. Liu, "Learning to Rank for Information Retrieval," *Foundations and Trends in Information Retrieval*, 2009, 3(3): 225–331.
- [27] E.J. Weyuker, T.J. Ostrand, and R.M. Bell, "Comparing the effectiveness of several modeling methods for fault prediction," *Empirical Software Engineering*, 2010, 15(3): 277–295.
- [28] K. Gao and T. M. Khoshgoftaar, "A comprehensive empirical study of count models for software defect prediction," *IEEE Transactions on Reliability*, 2007, 56(2): 223–236.
- [29] T. Joachims, "Optimizing search engines using clickthrough data," in: *ACM Proc. of KDD 2002, Canada*, 2012, pp. 133–142.
- [30] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I.H. Witten, "The WEKA Data Mining Software: An Update," *SIGKDD Explorations*, 2009, 11(1): 10–18.
- [31] Y. Freund, R. Iyer, R. Schapire, and Y. Singer, "An efficient boosting algorithm for combining preferences," *Journal of Machine Learning Research*, 2003, 4: 933–969.
- [32] M. Kendall, "A New Measure of Rank Correlation," *Biometrika*, 1938, 30(1–2): 81–89.

Stage-oriented Analysis on Factors Impacting Bug Fixing Time

Hong Wu^{1,2}, Junjie Wang¹, Qing Wang^{1,3}, Lin Shi¹, Feng Yuan^{1,4}

¹Laboratory for Internet Software Technologies, Institute of Software, Chinese Academy of Sciences, Beijing, China

²University of Chinese Academy of Sciences, Beijing, China

³State Key Laboratory of Computer Science, Institute of Software, Chinese Academy of Sciences, Beijing, China

⁴Institute of Software Application Technology, Guangzhou & Chinese Academy of Sciences, Guangzhou, China
{wuhong, wangjunjie, wq, shilin}@itechs.iscas.ac.cn, yf@gz.iscas.ac.cn

Abstract—The timely fixing of bugs is important to ensure software quality. In Open Source Software (OSS) development, behaviors of stakeholders impact the bug fixing process, especially the different stages respectively. However, most of the existing studies on impact factors of bug fixing time usually treat bug fixing process as a whole, while neglecting the particularity at its different stages. Ignoring the detail of different stages cannot let us understand why the fixing time is longer or shorter. In this paper, we aimed at investigating whether the factors have different impacts on the time of different stages and the whole process. Three stages of the whole fixing process were formalized, and twenty-four factors were defined and extracted from three aspects: bug reports, their associated source code and code changes. An empirical study based on two OSS projects, Eclipse JDT Core and Linux Kernel, was conducted for the investigation. The results of our study provide a very positive validation that the influence of factors on bug fixing time is stage related, rather than for the whole process. Our results can help developers better understand influences of factors on the bug fixing process, and thus provide opportunities to improve their process effectively.

Keywords—OSS maintenance; bug fixing time; stage-oriented analysis; empirical study

I. INTRODUCTION

Fixing bugs is an inevitable and time-consuming activity in the software development process. It is estimated that 80% of the total cost of a software system is spent on fixing bugs [3, 13]. Therefore, it is crucial to investigate the impact factors of bug fixing time, so as to effectively manage the fixing process.

In open source software communities, anyone who has interests in maintaining a software can participate in the bug fixing process, e.g., by posting comments to discuss the cause of a bug, or by taking an assignment of bug fixing, and etc. Behaviors of stakeholder usually change during the whole fixing process, and many stakeholders might only participate at certain stages of the whole process. In this case, stakeholders' behaviors impact the bug fixing process in an uncertain manner, especially at different stages of the process respectively. Therefore, the self-governing of developers in OSS results in the correlation between stakeholders' effort and the time of the whole bug fixing process may no longer be formed.

Prior studies have investigated the factors impacting bug fixing time, and proposed techniques to predict the time to fix a bug [1, 4, 7, 8, 11, 17, 19]. Most of them studied the time of the

bug fixing process as a whole (i.e., from a bug was reported until it was resolved), while neglecting the particularity at its different stages. However, some studies pointed out that factors with the same value can usually result in different fixing time [12]. This may derive from the inherent nature of the bug fixing process, i.e., the whole process consists of several individual stages, and different stages focus on different sub-activities of the whole process with different stakeholders involved.

Moreover, our observations on two large open source projects (i.e., Eclipse JDT Core and Linux Kernel) show that, along with different stakeholders' behaviors, the value of many factors are frequently changed during the whole fixing process. For instance, the assignee of bug #13939 in Eclipse JDT Core was changed 7 times. The summary of bug #86231 in Linux Kernel was modified 5 times. Due to the change in the value of these factors, their influence on individual stage might be different from that on the whole process. For example, the location of a bug has been considered as an impact factor of bug fixing time in prior study [8]. However, we found that, in Linux Kernel, the product category of a bug report has significant influence on the bug assignment stage and the bug fixing stage, rather than on the whole process. The above examples motivate this study.

In this paper, we presented a stage-oriented analysis of factors impacting bug fixing time. Three major stages of the bug fixing process were formalized, which are the bug assignment stage, the bug fixing stage and the bug verification stage. Twenty-four factors were defined and extracted from three aspects: bug report, the associated source code and code changes. We conducted an empirical study with three research questions as follows.

RQ1: Do the factors have different correlations with the time of different stages?

RQ2: Do the factors have different correlations between the time of individual stages and the time of the whole process?

RQ3: Which factors have the highest correlation with the time of each stage?

To answer the above three questions, we conducted the empirical study on two large open source projects (i.e., Eclipse JDT Core and Linux Kernel). The results revealed that factors have different influence among individual stage as well as the whole bug fixing process. For instance, the number of

comments and times of re-assignment have higher correlation with the time of the assignment stage than the time of the whole process. We believe our results can help developers better understand the influence of factors on the time of the bug fixing process, and illustrate the necessity of analyzing the influence of factors on bug fixing time by stages instead of the whole process.

Our study is different from prior work in that: we performed a stage-oriented analysis on the factors impacting bug fixing time, taking the change in the value of the factors into account. We believe these findings can provide new viewpoints and important references for efficient bug management, and provide an opportunity of improving current approaches of bug fixing time prediction for OSS.

The remainder of the paper is organized as follows. The setup of our study is described in Section II, and the results on two open source projects to answer our research questions are reported in Section III. The threats to validity are listed in Section IV. After summarizing the related work in Section V, we conclude our work in Section VI.

II. STUDY SETUP

In this section, we present the detailed design of our empirical study.

A. Subject Projects

The subject projects in our study are Eclipse JDT Core (*Eclipse* for short) and Linux Kernel (*Linux* for short). We choose these two projects for three reasons: (1) they are highly active projects and have been widely used in practice and prior studies; (2) they are from different domains (Integrated Development Environment vs. Operating System) and written in the two most famous and commonly used program languages (Java and C); (3) their development processes are well-managed with high-quality bug reports by Bugzilla and source code by Git.

B. Formalization of Bug Fixing Process

We first introduced four timepoints, and then defined three stages of the bug fixing process. Figure 1 visually presents the four timepoints and three stages in a timeline.

Bug Reporting Timepoint (T_R) is the timestamp when a bug is reported to Bugzilla by a user or developer.

Bug Assignment Timepoint (T_A) is the timestamp when a bug is assigned to the appropriate developer through Bugzilla. If reassignment occurred, we denote T_A as the timestamp when a bug is assigned to the developer who fixes the bug. Similar with previous work [17], changing assignee back to the default one ('xxx-inbox' in Eclipse or 'product-component' in Linux) is not considered as a bug assignment in our study.

Bug Fixing Timepoint (T_F) is the timestamp when the commits for fixing the bug are submitted to Git. If multiple commits are linked to one bug report, we use the submission time of the last commit as T_F .

Bug Verification Timepoint (T_V) is the timestamp when the status of a bug is marked as *VERIFIED*. We treat the time

when a bug is marked as *CLOSED* or *RESOLVED* as T_V in the case of the official *VERIFIED* status is missed out.

Based on these four timepoints, we defined the time of the whole bug fixing process as the interval between T_R and T_V , marked as *BLT* (standing for Bug Life Time). Moreover, we divided the bug fixing process into three main stages as follows.

Bug Assignment Stage (S_A) is the stage for understanding a bug report and assigning it to an appropriate developer for the fix. The time of S_A is defined as the interval between T_R and T_A , which is computed as $I_{AS} = T_A - T_R$.

Bug Fixing Stage (S_F) is the stage for the developer to fix the assigned bug by modifying the source code files. The time of S_F is defined as the interval between T_A and T_F , which is computed as $I_{FS} = T_F - T_A$.

Bug Verification Stage (S_V) is the stage for reviewers to verify the developers' resolution on the fix of assigned bugs. The time of S_V is defined as the interval between T_F and T_V , which is computed as $I_{VS} = T_V - T_F$.

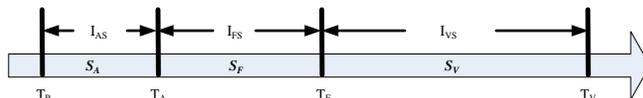


Figure 1. The stages of the bug fixing process and related timepoints

C. Definition of Factors

We defined 24 factors, which might influence the bug fixing time, from three aspects: 14 factors related to 8 attributes of bug reports, 3 factors related to the complexity of a bug fixing task measured by the source code, and 7 factors related to the effort required to fix a bug measured by the code changes. TABLE I shows the description of all defined factors. In particular, the scale of *ProdCat* and *CompCat* is nominal, the scale of *PriLevel* and *SevLevel* is ordinal, and the scale of the rest twenty factors is ratio.

D. Data Collection and Filtering

We collected data from bug reports, the associated source code and code changes to conduct our experiment, because we believe all of them contain the information that can influence bug fixing time for OSS. As we mentioned in Section II.A, for the two subject projects, bug reports were obtained from Bugzilla^{1,2}, while source code and code change were obtained from Git^{3,4}, respectively. The process of data collection and filtering is elaborated as follows.

Step 1: Retrieve commit logs and bug reports. We used `git log` to retrieve all commit logs from Git. For the collection of bug reports, we first obtained a set of bug IDs by using Bugzilla's search engine. In this study, we only collected IDs of fixed bug reports with the following search criteria: (1) the field of Status changed to *RESOLVED*, *VERIFIED* or *CLOSED* before Dec. 31, 2015; (2) the Resolution is marked as *FIXED* (in Eclipse) or *CODE_FIX* (in Linux). After removing

¹ <https://bugs.eclipse.org>

² <https://bugzilla.kernel.org>

³ [git://git.eclipse.org/gitroot/jdt/eclipse.jdt.core.git](https://git.eclipse.org/gitroot/jdt/eclipse.jdt.core.git)

⁴ [git://git.kernel.org/pub/scm/linux/kernel/git/stable/linux-stable.git](https://git.kernel.org/pub/scm/linux/kernel/git/stable/linux-stable.git)

the duplicated IDs in the above search results, we got a set of fixed bug IDs. Then we downloaded the webpage of the bug report associated with each bug ID. In this way, we obtained a set of bug reports.

Step 2: Link commit logs to bug reports to obtain a preliminary dataset. To establish the linkage between bug reports and its corresponding fixing commit, we first identify the bug IDs appearing in the commit logs by two patterns: (1) for Eclipse, bug IDs are included in the subject of commit logs; (2) for Linux, bug IDs are appeared in the content of commit logs with the form of the bug's URL in Bugzilla. If a bug ID was identified in the commit log, we compared it with the set of bug reports obtained in Step 1. If matched, the commit was considered as a bug-fixing commit dedicated to the corresponding bug report. Thus we included a pair of bug report and commit logs as a data item in the preliminary dataset.

TABLE I. SUMMARY OF DEFINED FACTORS

Aspect	Factor	Description
Bug Reports	ProdCat	The product category of a bug
	CompCat	The component category of a bug
	PriLevel	The priority level of a bug
	SevLevel	The severity level of a bug
	LenSum	The length of the summary of a bug report in terms of English words
	LenDesc	The length of the description of a bug report in terms of English words
	NumCom	The number of comments for the bug
	LenCom	The length of total comments of a bug report in terms of English words
	CHSum	The number of changes for the summary content of a bug
	CHProd	The number of changes for the product category of a bug
	CHComp	The number of changes for the component category of a bug
	CHPri	The number of changes for the priority level of a bug
	CHSev	The number of changes for the severity level of a bug
CHAssi	The number of changes for the assignee of a bug	
Source Code	SLOC	The number of source lines of code of all changed files
	NumMethod	The number of methods of all changed files
	Method_CC	The cycomatic complexity of methods of all changed files
Code Changes	NumCFile	The number of all changed files
	AddSLOC_M	The number of added source lines within methods
	AddSLOC_F	The number of added SLOC for fields
	AddLOC_C	The number of added lines of changed comments
	DelSLOC	The number of deleted SLOC for changed code
	NumCMethod	The number of all changed methods
	CMethod_CC	The cycomatic complexity of all changed methods

Step 3: Filter invalid data items to obtain a filtered dataset. We filtered out certain invalid data items from the preliminary dataset, and thus a filtered dataset was obtained. The following exclusive criteria were used for filtering.

Data items that contain bug report whose severity is enhancement: According to the regulation of Bugzilla, this kind of bug reports is actually the feature request. We filtered them out from the preliminary dataset, because they are out of the scope of our study.

Data items that contain bug report which was reopened: We found about 12% data items in both subject projects contained reopened bug reports. Such data items often involved a more complex fixing process, which is quite different from the majority of the data items. Therefore, we filtered out these data items for drawing more general conclusions.

Data items that contain commits which have no source code changes: We filtered out the data items which contain commits with configuration files, pictures and etc., because the factors of source code and code changes cannot be extracted from such data items.

Data items with negative time of the stage of bug fixing process: About 10% data items in Eclipse and 20% in Linux have negative value of I_{FS} or I_{VS} . This may be caused by several reasons. For instance, the developer, who is both the bug triager and the bug fixer, starts to fix a bug and submit commits before he assigns the bug report to himself (e.g., bug #50781 in Linux), or the developer changes the bug report status prior to submitting the fixing commits (e.g., bug #140879 in Eclipse). We filtered out those data to ensure the validity of our outcome.

Step 4: Retrieve source code and code change files to obtain the final dataset. For each data item in the filtered dataset, based on its commit logs, we used `git show` to get the detailed code changes of each commit. Moreover, for each commit, we used `git checkout` to get the source code files with the post-fix version and change history logs of those files. Then we get the commit ID in the pre-fix version of each changed source code file from its change history log, and we used `git checkout` again to get the source code files in the pre-fix version. In our study, we defined the pre-fix version of source code file as the first previous version that has diffs with the post-fix version. We used the current version (i.e., post-fix version) for a new added source code file, because there is no pre-fix version for new added source code files in Git.

Finally, we obtained 5493 data items, including 4297 bug reports linked with 4984 commits from Eclipse, and 1196 bug reports linked with 1272 commits from Linux. In particular, for each commit, we obtained its detailed change log, a set of changed source code files in the pre-fix version and those in the post-fix version. Due to the page limitation, we present the detailed statistics of the data set in our project webpage⁵, and also make the data set available.

E. Computation of Factor Value

For factors related with bug reports, we extracted their values from the downloaded webpages. In particular, we parsed the modification history of the bug report, with each modification record (*MR*) including who, when, removed or added what. We divided *MRs* into S_A , S_F and S_V according to its timestamp as well as the timestamp of each stage illustrated in Section II.B, and then we got the value of each factor in each

⁵ http://itechs.iscas.ac.cn/membersHomepage/wuhong/seke_project.html

defined stage and for the whole bug fixing process. For factors related with source code, we applied Lizard⁶ to parse the source code files in the pre-fix version and compute the value of these factors. For factors related with code changes, first, we developed a tool to parse the detailed commit change log to obtain the index of changed lines of each source code file in the commit. Then we applied the tool Lizard again to parse the source code files in the post-fix version to get their detailed source code structure. Based on the above results, we computed the value of factors related with code changes.

F. Analysis Methods

In order to investigate the relationship between the defined factors and the time of different stages (i.e., I_{AS} , I_{FS} and I_{VS}) as well as the time of the whole process (BLT), two different tests were performed based on the scale of the factors for two subject projects respectively.

For nominal and ordinal factors, we used Kruskal-Wallis test [14] to examine whether there are significant differences in bug fixing time among different values of each factor. Taking $PriLevel$ as an example, we first divided the data items into different groups according to their priority levels (e.g. there are five priority levels in Eclipse), then we extracted the time of corresponding data items for each individual stage as well as the whole process. Kruskal-Wallis test was performed among these groups of data items. P-value < 0.05 denotes that there exist significant differences among the time of data items with different priority levels, which might indicate that the factor of priority level has significant influence on bug fixing time.

For ratio factors, we used Spearman rank correlation test [16] to examine whether there are correlations between the value of certain factors and the bug fixing time. Taking $NumCom$ as an example, we first established two groups of data, one group corresponding to the value of $NumCom$ of each data item, and the other group corresponding to its bug fixing time. Then we performed Spearman rank correlation test for these two groups. P-value < 0.05 denotes $NumCom$ correlates with bug fixing time significantly, further reflecting that it has significant influence on bug fixing time.

III. RESULTS AND ANALYSIS

In this section, we report the results and analysis on the three research questions. As mentioned in Section II.F, the results of Kruskal-Wallis test for nominal and ordinal factors and Spearman rank correlation test for ratio factors are presented in TABLE II. We used ‘-’ to denote factors whose correlation is greater than 0.05 and use ‘n/a’ to denote the factors that are not applicable for the test.

A. *RQ1: Do the factors have different correlations with the time of different stages?*

It is quite common that the factors have different correlations with the time of different stages. From TABLE II, we can observe that all the ratio factors have different correlations with the time of different stages, and many of these factors have large different correlations with the time among

stages, e.g., the correlations between $NumCom$ and the time of each stage are 0.873, 0.390 and 0.081 respectively in Linux.

Moreover, we can observe that the value of correlations in S_A is the highest in general, while the value of correlations in S_V is the lowest or even absent. For instance, for both subject projects, factors related with comments ($NumCom$ and $LenCom$) have higher correlations with I_{AS} (0.541 and 0.531 in Eclipse) and I_{FS} (0.361 and 0.368 in Eclipse) than with I_{VS} (0.204 and ‘-’ in Eclipse). Another example is the factors related with code changes. They have correlations with I_{FS} , but most of their correlations with I_{VS} are quite low or even absent. This is also hold good for the nominal and ordinal factors. For instance, for both subject projects, $CompCat$ and $PriLevel$ have no significant influence on I_{VS} , while their influences on I_{AS} are usually significant.

These findings indicate that the influences of factors on bug fixing time are stage related, and during the bug fixing process, the influence of the factors on bug fixing time weaken over time. Moreover, the results verify the necessity of adopting a stage-oriented method for analyzing the relationship between factors and the bug fixing time.

B. *RQ2: Do the factors have different correlations between the time of individual stages and the time of the whole process?*

It is quite different between the correlations of the factors with the time of stages and those of the whole process. We can observe from TABLE II that, some factors which have no correlations with BLT are actually correlated with certain stages, e.g., source code related factors in both subject projects. Moreover, most of the factors have higher correlations with I_{AS} or I_{FS} than with BLT , e.g., $CHProd$, $CHComp$ and $CHAssi$ in both subject projects. This can be understood that a wrong assignment of location and assignee of bug reports could prolong the time of the assignment stage. However, when putting them to the whole process, their influences become much weaker.

These findings might imply that the influence of the factors on bug fixing time could mainly exist on S_A and S_F , which reflects a more accurate relationship between the factors and the time of the bug fixing process. In addition, the results verify the necessity of adopting a stage-oriented analysis again.

C. *RQ3: Which factors have the highest correlation with the time of each stage?*

In TABLE II, we highlighted the top three correlations for each stage (only one for S_V due to the few and low correlation results), and we can make the following observations.

In S_A , the factors with top three highest correlations are $NumCom$, $LenCom$ and $CHAssi$ in both subject projects. Let’s first focus on $NumCom$. The high correlation value (0.541 in Eclipse and 0.873 in Linux) means the number of comments could influence the time of the assignment stage. People might expect that more comments signify there is more attention focused on the bug, which should help find the right bug fixer, and thus it results in a shorter assignment time. However, in reality, bug reports with more comments may accompany with

⁶ <http://www.lizard.ws/>

TABLE II. TEST RESULTS FOR THE IMPACT OF FACTORS ON BUG FIXING TIME

Factor	Eclipse				Linux			
	I_{AS}	I_{FS}	I_{VS}	BLT	I_{AS}	I_{FS}	I_{VS}	BLT
<i>P-value of Kruskal-Wallis test for the nominal and ordinal factor</i>								
ProdCat	-	-	-	n/a	0.000	0.015	-	-
CompCat	0.000	-	-	n/a	0.000	0.000	-	0.003
PriLevel	0.000	-	-	0.041	0.000	0.000	-	0.000
SevLevel	0.000	0.000	0.000	0.000	-	-	-	-
<i>Correlation values of Spearman rank correlation test for the ratio factor (p-value = 0.05)</i>								
LenSum	-	0.078	-0.024	0.056	-	-	-	-
LenDesc	0.203	0.091	0.026	0.075	-	0.065	-	0.070
NumCom	0.541	0.361	0.204	0.266	0.873	0.390	0.081	0.139
LenCom	0.531	0.368	-	0.215	0.871	0.357	0.129	0.143
CHSum	0.261	0.131	-	0.181	0.386	0.081	0.061	0.066
CHProd	0.139	-	-	0.061	0.325	0.161	-	0.075
CHComp	0.304	-	-	0.113	0.413	0.153	-	0.102
CHPri	0.087	0.086	-	0.046	0.091	-	-	-
CHSev	0.100	0.036	-	0.035	0.161	-	-	-
CHAssi	0.628	-	-	0.208	0.958	-	-	0.094
SLOC	n/a	0.140	-	-	n/a	-	-	-
NumMethod	n/a	0.133	0.036	-	n/a	0.084	-	-
Method_CC	n/a	0.131	-	-	n/a	0.058	-	-
NumCFile	n/a	0.134	0.032	-	n/a	0.172	-	0.087
AddSLOC_M	n/a	0.239	-	-	n/a	0.203	-	0.109
AddSLOC_F	n/a	0.089	-	-	n/a	0.156	-	0.138
AddLOC_C	n/a	0.210	-	-	n/a	0.135	-	0.070
DelSLOC	n/a	0.115	0.030	-	n/a	0.085	-	-
NumCMethod	n/a	0.179	-	-	n/a	0.183	-	0.094
CMethod_CC	n/a	0.160	-	-	n/a	0.130	0.061	0.082

a longer assignment time, especially for Linux. This might be because that more comments indicate the bug is difficult to fix, which brings in more people to discuss the resolution in comments. Furthermore, more comments would call for more time and effort from developers to read and make final decisions, which could also potentially extend the bug assignment time. The results in TABLE II also reveal that the number of changes in assignee could influence I_{AS} . This is obvious that the change of assignee, commonly known as bug tossing [2, 6], would prolong the bug assignment time. Hence, distributing bug reports to appropriate fixers quickly and precisely can effectively shorten the bug assignment time.

In S_F , the factors with top three highest correlations are *NumCom*, *LenCom* and *AddSLOC_M* in both subject projects. Number and length of comments can influence I_{FS} due to similar reason as mentioned above. Moreover, *AddSLOC_M* has correlation with I_{FS} , which is under our expectation because the more coding effort in terms of number of added source code can easily result in a longer bug fixing time.

In S_V , there are almost no correlations between the defined factors and I_{VS} . However, we also found that, in the two subject projects, I_{VS} can occupy more than 50% of BLT . To investigate the reason for this phenomenon, we randomly sampled 10% bug reports for each project, and manually examined their contents. We found that, almost all the sampled bug reports in both projects changed their status to *VERIFIED* on the day when the associated version released or a new release tag was assigned. Put another way, the time of verification stage cannot precisely reflect the situation of bug verification activity. That's why there are few factors that have correlation with the long I_{VS} in both projects.

IV. THREATS TO VALIDITY

In this section, we discuss the threats to validity of our study with respect to construct validity, internal validity and external validity [16].

Construct Validity: The factors used in our study are generally well understood and straightforward to compute based on publicly available datasets of two OSS projects, which enable the replication of this study. Therefore, our study can achieve a strong confidence in construct validity.

Internal Validity: In our study, we relied on the information stored in Bugzilla and Git repositories to construct the link between bug reports and commits. The treatment can obtain precise links at the cost of filtering some bug reports without such a link. This may influence the internal validity. We note that there are techniques to recover the missing link between bug reports and commit (e.g., [20]). In the future, we would like to employ such techniques to help find more links, and further minimize this threat.

External Validity: The subject projects used in our study are highly active in open source community and have been widely used in previous work. Moreover, they are of different domains and use different development languages. However, we have used only two projects, which might make our findings not generalizable enough to other open source projects. This risk could be mitigated by adding more subject projects. This will be explored in our future work.

V. RELATED WORK

A lot of researches have been conducted to empirically investigate the impact factors of bug fixing time. Mockus et al.

[9] found that in Apache and Mozilla, bugs with higher priority were fixed faster. Panjer [11] found that, in Eclipse project, bugs with little discussion tend to be resolved quickly, however, when bugs receive more conversation, the resolution times become dependent on their severity level. Marks et al. [8] studied bug fixing time in Eclipse and Mozilla, and found that the time taken to report a bug and its location have the most impact on bug fixing time. Anbalagan and Vouk [1] studied the bug reports of Ubuntu project and found that there is a strong linear relationship between the number of users participating in a bug report and the median time taken to fix it. Besides the attribute of bug reports, Saha et al. [12] extracted code change metrics, e.g., number of changed files, for analyzing the reason of long live bugs in four Eclipse projects. Hooimeijer and Weimer [5] measured bug-report-triage time using regression analysis based on bug report metrics. Zhang et al. [17] investigated impact factors in order to understand why delays incurred during bug fixing. These prior researches treated bug fixing as a whole process, or only focused on a particular phase of the process. An ignored phenomenon is that bug fixing is multi-stage process, in which case the influence of factors cross the whole fixing process might not remain the same. In contrast to prior researches, our work performed a stage-oriented analysis to explore such situation.

VI. CONCLUSIONS

In this paper, we performed a stage-oriented analysis to investigate the factors impacting bug fixing time based on two large OSS projects. We extracted twenty-four factors from three aspects: bug reports, their associated source code and code changes, and empirically investigated the influence of them on the time of three stages of the bug fixing process and the whole process. Our results show that the influences of factors on bug fixing time are stage related, and thus it is necessary to analyze the relationship between factors and the bug fixing time in a stage-oriented way. We believe our findings can help developers better understand impact factors on bug fixing time, and thus improve bug fixing process management effectively.

In the future, we plan to study more data sources from more projects (both OSS projects and industry oriented projects), and investigate more factors extracted from new dimensions (e.g., participants and code review activities) in order to make more generic findings.

ACKNOWLEDGMENT

This work is sponsored by the NSFC under Grant No. 91218301, 91318302 and 61432001, and this work is also sponsored by the FoShanCAS under Grant No. 2014HT100022.

REFERENCES

- [1] P. Anbalagan, and M. Vouk, "On predicting the time taken to correct bug reports in open source projects," In ICSM'09: Proceeding of IEEE International Conference on Software Maintenance, pp. 523–526, September 2009.
- [2] P. Bhattacharya, and I. Neamtiu, "Fine-grained incremental learning and multi-feature tossing graphs to improve bug triaging," In ICSM'10: Proceedings of IEEE International Conference on Software Maintenance, pp. 1–10, September 2010.
- [3] J. Frederick P. Brooks, *The Mythical Man-month* (Anniversary Ed.), Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1995.
- [4] P. J. Guo, T. Zimmermann, N. Nagappan, and B. Murphy, "Characterizing and predicting which bugs get fixed: An empirical study of Microsoft Windows," In ICSE'10: Proceedings of the 32nd ACM/IEEE International Conference on Software Engineering, pp. 495–504, May 2010.
- [5] P. Hooimeijer, and W. Weimer, "Modeling bug report quality," In ASE'07: Proceedings of the Twenty-second IEEE/ACM International Conference on Automated Software Engineering, pp. 34–43, November 2007.
- [6] G. Jeong, S. Kim, and T. Zimmermann, "Improving bug triage with bug tossing graphs," In ESEC/FSE'09: Proceedings of the 7th joint meeting of the European software engineering conference and the ACM SIGSOFT symposium on The Foundations of Software Engineering, pp. 111–120, August 2009.
- [7] S. Kim, and J. E. James Whitehead, "How long did it take to fix bugs?" In MSR'06: Proceedings of the 2006 International Workshop on Mining Software Repositories, pp. 173–174, May 2006.
- [8] L. Marks, Y. Zou, and A. E. Hassan, "Studying the fixtime for bugs in large open source projects," In Promise'11: Proceedings of the 7th International Conference on Predictive Models in Software Engineering, pp. 11:1–11:8, September 2011.
- [9] A. Mockus, R. T. Fielding, and J. D. Herbsleb, "Two case studies of open source software development: Apache and mozilla," *ACM Transaction Software Engineering Methodology*, 11(3):309–346, July 2002.
- [10] H. Naguib, N. Narayan, B. Brugge, and D. Helal, "Bug report assignee recommendation using activity profiles," In MSR'13: Proceedings of the 10th IEEE Working Conference on Mining Software Repositories, pp. 22–30, May 2013.
- [11] L. D. Panjer, "Predicting eclipse bug lifetimes," In MSR'07: Proceedings of the Fourth International Workshop on Mining Software Repositories, pp. 29–32, May 2007.
- [12] R. K. Saha, S. Khurshid, and D. E. Perry, "Understanding the triaging and fixing processes of long lived bugs," *Information and Software Technology*, 65:114–128, September 2015.
- [13] L. Tan, C. Liu, Z. Li, X. Wang, Y. Zhou, and C. Zhai, "Bug characteristics in open source software," *Empirical Software Engineering*, 19(6):1665–1705, 2014.
- [14] E. Theodorsson-Norheim, "Kruskal-wallis test: Basic computer program to perform nonparametric one-way analysis of variance and multiple comparisons on ranks of several independent samples," *Computer Methods and Programs in Biomedicine*, 23(1):57–62, August 1986.
- [15] Y. Tian, D. Lo, and C. Sun, "Information retrieval based nearest neighbor classification for fine-grained bug severity prediction," In WCRE'12: Proceedings of the 19th Working Conference on Reverse Engineering, pp. 215–224, October 2012.
- [16] C. Wohlin, R. Runeson, M. Host, M. C. Ohlsson, B. Regnell, and A. Wesslen, *Experimentation in Software Engineering*, Springer Science and Business Media, 2012.
- [17] F. Zhang, F. Khomh, Y. Zou, and A. E. Hassan, "An empirical study on factors impacting bug fixing time," In WCRE'12: Proceedings of the 19th Working Conference on Reverse Engineering, pp. 225–234, October 2012.
- [18] T. Zhang, and B. Lee, "How to recommend appropriate developers for bug fixing?" In COMPSAC'12: Proceedings of the 36th IEEE Annual Computer Software and Applications Conference, pp. 170–175, July 2012.
- [19] C. Weiss, R. Premraj, T. Zimmermann, and A. Zeller, "How long will it take to fix this bug?" In MSR'07: Proceedings of the Fourth International Workshop on Mining Software Repositories, pp. 1–8, May 2007.
- [20] R. Wu, H. Zhang, S. Kim, and S.-C. Cheung, "Relink: Recovering links between bugs and changes," In ESEC/FSE'11: Proceedings of the 19th ACM SIGSOFT Symposium and the 13th European Conference on Foundations of Software Engineering, pp. 15–25, September 2011.

Heterogeneous Defect Prediction via Exploiting Correlation Subspace

Ming Cheng¹, Guoqing Wu¹, Min Jiang², Hongyan Wan¹, Guoan You¹ and Mengting Yuan¹

1. State Key Laboratory of Software Engineering, School of Computer, Wuhan University, Wuhan 430072, China
{chengming, wgq, why0511, 2014202110064, ymt}@whu.edu.cn

2. Department of Cognitive Science and Technology and Fujian Key Laboratory of Brain-Like Intelligent Systems, Xiamen University, Xiamen 361005, China
minjiang@xmu.edu.cn

Abstract—Software defect prediction generally builds models from intra-project data. Lack of training data at the early stage of software testing limits the efficiency of prediction in practice. Thereby researchers proposed cross-project defect prediction using the data from other projects. Most previous efforts assumed the cross-project defect data have the same metrics set which means the metrics used and size of metrics set are same in the data of projects. However, in real scenarios, this assumption may not hold. In addition, software defect datasets have the class imbalance problem increasing the difficulty for the learner to predict defects. In this paper, we advance canonical correlation analysis for deriving a joint feature space for associating cross-project data and propose a novel support vector machine algorithm which incorporates the correlation transfer information into classifier design for cross-project prediction. Moreover, we take different misclassification costs into consideration to make the classification inclining to classify a module as a defective one, alleviating the impact of imbalanced data. Experiments on public heterogeneous datasets from different projects show that our method is more effective, compared to state-of-the-art methods.

Keywords—defect prediction; heterogeneous metrics; class imbalance; canonical correlation analysis; support vector machine

I. INTRODUCTION

Recently, most effective software defect prediction approaches have been proposed and attracted a lot of attention from academic and industrial communities. Most prior studies generally focused on Within-Project Defect Prediction (WPDP), which trained prediction model from historical data to detect the defect proneness of new software modules within the same project [1][2][3][4]. However, researchers often confront the situations that there are not enough historical data available, and they have to resort to the data from other projects to aid the learning of the target projects. Owing to this reason, Cross-Project Defect Prediction (CPDP) is proposed. It is the art of using data of inter-project to predict software defects in the target project with a small amount of local data [5][6].

Existing CPDP approaches are based on the underlying assumption that both source and target project data should exhibit the same data distribution or are drawn from the same feature space (i.e., the same software metrics). When the distribution of the data changes, or when the metrics features for source and target projects are different, one cannot expect the resulting prediction performance to be satisfactory. We

consider this scenarios as Heterogeneous Cross-Project Defect Prediction (HCPDP) [7][8].

Mostly, the software defect data sets are imbalanced, which means the number of the defective modules is usually much smaller than that of the defective-free modules [9][10]. The imbalanced nature of data can cause poor prediction performance. That is, the probability of defect prediction can be low while the overall performance is high. Without taking this issue into account, the effectiveness of software defect prediction in many real-world tasks would be greatly reduced.

Recently, some researchers have noticed the importance of these problems in software defect prediction. For example, Nam et al. [7] used the metrics selection and metrics matching to select similar metrics for building a prediction model with heterogeneous metrics set. They discarded dissimilar metrics, which may contain useful information for training. Jing et al. [8] introduced Canonical Correlation Analysis (CCA) into HCPDP, by constructing the common correlation space to associate cross-project data. Then, one can simply project the source and target project data into this space for defect prediction. Like previous CPDP methods, the class imbalance problem of software defect datasets was not taken into account. Ryu et al. [10] designed the Value-Cognitive Boosting with Support Vector Machine algorithm (VCB-SVM) which exploited sampling techniques to solve the class imbalance issue for cross-project environments. Nevertheless, sampling strategy alters the distribution of the original data, where it may discard some potentially useful samples that could be important for prediction process. Therefore, these methods are not good solutions for addressing the class imbalance issue under heterogeneous cross-project environments.

Motivated by these observations, we advance CCA for driving a joint feature space for associating cross-project data, and incorporate the correlation transfer information in the derived subspace for improved prediction performance. During the defect prediction process, the two types of misclassification errors are encountered. Type I misclassifies a defective-free module as defective one (increasing the development cost), while type II misclassifies a defective module as defective-free one (leading to the more risk cost). Hence, we can take different misclassification costs into consideration by incorporating the cost factors into the SVM model. Fig. 1 shows a summary of our method.

In this paper, we propose a novel Cost-sensitive Correlation Transfer Support Vector Machine (CCT-SVM) algorithm to

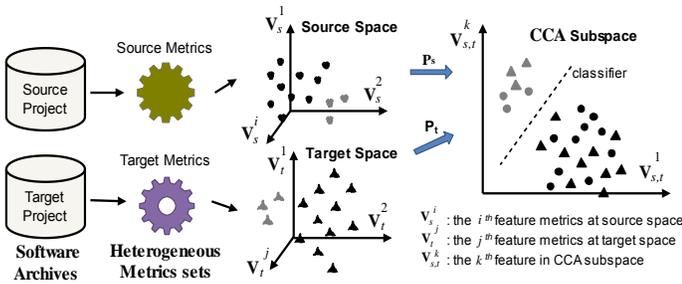


Figure 1. The overview of heterogeneous cross-project prediction

deal with the class imbalance problem under HCPDP settings. Our contributions to the current state of research are summarized as follows.

- We advance CCA for deriving a joint feature space to associate cross-project data, so the correlation transfer information can be exploited to improve the prediction performance.
- During the software defect prediction process, we emphasize the risk cost to make the classification inclining to classify a module as a defective one, alleviating the impact of imbalanced data.
- Conduct an extensive and large-scale empirical study to evaluate our method.

II. RELATED WORK

A. Cross-project Defect Prediction

Software defect prediction employs historical data on reported (or repaired) defects to predict the location of previously unknown defects that hide in the code. However, sufficient historical training data from the same project is not always available or is hard to collect in practice. To address this issue, researchers proposed CPDP as an alternative solution in the last few years [11] [12] [13].

Zimmermann et al. [11] evaluated cross-project prediction performance based on large-scale experiments for 12 real-world projects. It indicated that current CPDP models do not perform well in most cases. Similar to Zimmermann's work, He et al. [12] investigated CPDP by focusing on training data selection. They pointed out that the prediction performance was related to the distributional attributes of datasets. Turhan et al. [13] proposed a Nearest-Neighbor filter method (NN-filter) to select similar data from source project. They only used nearest neighbors for each test data to construct training set, which have similar features to local data. Unlike the prior work selecting training data which are similar from the test data, Ma et al. [5] proposed Transfer Naive Bayes (TNB), by using the information of all the proper features in training data. The TNB transferred cross-project data information into the weights of the training data. Based on these weighted data, the prediction model was built. Nam et al. [6] applied Transfer Component Analysis (TCA) to CPDP and proposed TCA+ by selecting a suitable normalization automatically to preprocess data.

However, existing CPDP approaches are based on the assumption that the source and target data have the same software metrics set. When the metrics features for source and target projects are different, these methods will be unavailable. Recently, Nam et al. [7] proposed the Heterogeneous Defect Prediction (HDP) to predict defects across projects with

heterogeneous metrics sets. They conducted metrics selection and metrics matching to build a prediction model using 28 projects collected from public defect datasets. Jing et al. [8] introduced CCA into CPDP to make the data distributions of source and target projects similar. They extended CCA to CCA+, by unified metric representation technique to preprocess data, so that the correlation between the projected data in CCA space is maximized.

Previous work proposed various defect prediction models under heterogeneous cross-project settings, but the class imbalance issue of defect datasets was not taken into account. Hall et al. [14] pointed out that data imbalance with regard to specific classification approaches may produce poor performance. Ignoring this issue, a learner that minimizes the prediction error would often produce a useless predictive model that predicts all the modules as defect-free.

B. Class Imbalance Learning

Class imbalance learning refers to learning from data that exhibit significant imbalance among classes. The challenge of class imbalance is that relatively underrepresented class cannot draw equal attention to the learning algorithm compared with the majority class, which often leads to poor prediction performance [15]. To achieve better sensitivity to the minority class, the class imbalance issue should be explicitly tackled.

Wang et al. [15] explored the impact of class imbalance issue and provided guidance and valuable information for designing good predictor for software defects. Zheng et al. [16] employed three cost-sensitive boosting neural network algorithms for software defect prediction and found that threshold-moving algorithm was the best. Grbac et al. [17] studied the performance of machine learning techniques with different level of imbalance for software defect data. The feature selection and data sampling were exploited together. This method addressed class imbalance by modifying the training data. Ren et al. [18] proposed kernel based prediction method to address the class imbalance. The NASA and SOFTLAB datasets were used for experiments. Sun et al. [19] presented a coding based ensemble learning method, which converted class imbalance data into balanced multiclass data with specific coding scheme to avoid the class imbalance problem. Ryu et al. [10] first investigated whether the class imbalance learning can improve the prediction performance under CPDP settings. And they designed the value-cognitive boosting with support vector machine algorithm dealing with the class imbalance issue for cross-project environments. Experimental results showed that the class imbalance learning can be beneficial for CPDP.

III. HETEROGENEOUS SOFTWARE DEFECT PREDICTION

In this section, we present our method for HCPDP, which includes two parts: unified metrics representation and the CCT-SVM model.

A. Unified Metric Representation for Heterogeneous Data

To effectively utilize the heterogeneous metrics features from cross-project data, Jing et al. [8] proposed a Unified Metric Representation (UMR) to make the heterogeneous data can be compared. Based on the UMR, the standard CCA was exploited to find a common space for data from source and target project such that the correlation between the projected data in that space was maximized.

Similar to [8], we also exploit the UMR technique to make heterogeneous data to be compared. Suppose that $\mathbf{X}_s = \{\mathbf{x}_s^1, \mathbf{x}_s^2, \dots, \mathbf{x}_s^N\} \in \mathbb{R}^{d_s \times N}$ and $\mathbf{X}_t = \{\mathbf{x}_t^1, \mathbf{x}_t^2, \dots, \mathbf{x}_t^M\} \in \mathbb{R}^{d_t \times M}$ separately denote the source and target project data, where \mathbf{x}_s^i indicates the i^{th} model in \mathbf{X}_s , N and M denote the numbers of modules in \mathbf{X}_s and \mathbf{X}_t , respectively. $\mathbf{x}_s^i = [a_s^{i1}, a_s^{i2}, \dots, a_s^{id_s}]$ and $\mathbf{x}_t^i = [a_t^{i1}, a_t^{i2}, \dots, a_t^{id_t}]$ represent a module in the source and target project, where a_s^{ij} indicates the j^{th} metrics feature of the i^{th} model in source project, d_s and d_t are the numbers of metrics in \mathbf{X}_s and \mathbf{X}_t , $d_s \neq d_t$. Here, we exploit UMR to restructure data as follows:

$$\bar{\mathbf{X}}_s = \begin{bmatrix} \mathbf{X}_s^c \\ \mathbf{X}_s^s \\ 0_{(d_t-d_c) \times N} \end{bmatrix}, \quad \bar{\mathbf{X}}_t = \begin{bmatrix} \mathbf{X}_t^c \\ \mathbf{X}_t^t \\ 0_{(d_s-d_c) \times M} \end{bmatrix} \quad (1)$$

where the \mathbf{X}_s^c and \mathbf{X}_t^c are the same common metrics, \mathbf{X}_s^s and \mathbf{X}_t^t are specific metrics in source and target project, respectively. Note that if there exist no common metrics, then $\mathbf{X}_s^c = \mathbf{X}_t^c = 0$.

B. Learning Correlation Subspace via CCA

Based on the obtained UMR for heterogeneous data, we employ CCA technique to determine a common representation (e.g. a joint subspace) for features extracted from source and target projects, so that the model trained in the source project can be applied to detect the test modules in the target project. CCA learns two projection vectors $\mathbf{p}_s \in \mathbb{R}^{d_s}$ and $\mathbf{p}_t \in \mathbb{R}^{d_t}$, which maximize the following linear correlation coefficient ρ :

$$\max_{\mathbf{p}_s, \mathbf{p}_t} \rho = \frac{\mathbf{p}_s^T \Sigma_{st} \mathbf{p}_t}{\sqrt{\mathbf{p}_s^T \Sigma_{ss} \mathbf{p}_s} \sqrt{\mathbf{p}_t^T \Sigma_{tt} \mathbf{p}_t}} \quad (2)$$

where Σ_{ss} and Σ_{tt} represent the within-project covariance matrices of $\bar{\mathbf{X}}_s$ and $\bar{\mathbf{X}}_t$ respectively, while $\Sigma_{st} = \Sigma_{ts}$ represents the cross-project covariance matrix of $\bar{\mathbf{X}}_s$ and $\bar{\mathbf{X}}_t$. Σ_{ss} , Σ_{tt} and Σ_{st} are separately defined as:

$$\Sigma_{ss} = \frac{1}{N} \sum_{i=1}^N (\bar{\mathbf{x}}_s^i - \mathbf{m}_s)(\bar{\mathbf{x}}_s^i - \mathbf{m}_s)^T \quad (3)$$

$$\Sigma_{tt} = \frac{1}{M} \sum_{i=1}^M (\bar{\mathbf{x}}_t^i - \mathbf{m}_t)(\bar{\mathbf{x}}_t^i - \mathbf{m}_t)^T \quad (4)$$

$$\Sigma_{st} = \frac{1}{NM} \sum_{i=1}^N \sum_{j=1}^M (\bar{\mathbf{x}}_s^i - \mathbf{m}_s)(\bar{\mathbf{x}}_t^j - \mathbf{m}_t)^T \quad (5)$$

where $\bar{\mathbf{x}}_s^i$ represents the i^{th} software module in $\bar{\mathbf{X}}_s$, \mathbf{m}_s and \mathbf{m}_t are the mean modules of $\bar{\mathbf{X}}_s$ and $\bar{\mathbf{X}}_t$. As proved in [8], the optimization of (2) can be solved as a generalized eigenvalue decomposition problem:

$$\begin{pmatrix} 0 & \Sigma_{st} \\ \Sigma_{st} & 0 \end{pmatrix} \begin{pmatrix} \mathbf{p}_s \\ \mathbf{p}_t \end{pmatrix} = \lambda \begin{pmatrix} \Sigma_{ss} & 0 \\ 0 & \Sigma_{tt} \end{pmatrix} \begin{pmatrix} \mathbf{p}_s \\ \mathbf{p}_t \end{pmatrix} \quad (6)$$

The λ is the generalized eigenvalue corresponding to the generalized eigenvector $\begin{pmatrix} \mathbf{p}_s \\ \mathbf{p}_t \end{pmatrix}$. Generally, we can derive more than one pair of canonical components $\{\mathbf{p}_s^k\}_{k=1}^{d_v}$ and $\{\mathbf{p}_t^k\}_{k=1}^{d_v}$ with corresponding ρ_i in a descending order ($\rho_i > \rho_{i+1}$). Note that d_v is the dimension number of the correlation subspace of CCA. We can construct the projective transformation matrices $\mathbf{P}_s = [\mathbf{p}_s^1, \dots, \mathbf{p}_s^{d_v}] \in \mathbb{R}^{d_s \times d_v}$ and $\mathbf{P}_t = [\mathbf{p}_t^1, \dots, \mathbf{p}_t^{d_v}] \in \mathbb{R}^{d_t \times d_v}$.

Once the correlation subspace is derived, the test modules at target project can be directly detected by the model learned from the source project data projected onto the subspace.

C. Cost-sensitive Correlation-transfer SVM

In derived CCA subspace, each dimension $\mathbf{v}_{s,t}^i$ is associated with a different correlation coefficient ρ_i . A higher ρ_i denotes a better correlation, which results in a better transfer ability for the associated dimension $\mathbf{v}_{s,t}^i$. On the other hand, poorer transfer ability will increase classification error, even the classifier is trained using the projected source project data. Obviously, higher correlation coefficient can obtain more class discriminant information which is more useful for constructing classifiers [20].

In SVM, if the i^{th} feature attribute has the better discrimination ability, the classical SVM could produce a larger magnitude for the corresponding model (e.g., a larger $|w_i|$). Here, we introduce a correlation regularizer and propose a linear SVM model which integrates the cross-project transfer ability and class discrimination in a unified formulation. Moreover, we employ a specific misclassification costs to minimize a classification-oriented loss. We set two misclassification cost values C^+ and C^- . C^+ is the misclassification cost for the defective modules, while C^- is the misclassification cost for the defective-free modules. By assigning a higher misclassification cost for the minority defective modules than the majority defective-free modules (i.e., $C^+ > C^-$), the effect of class imbalance could be reduced. Then, the modified SVM decision function can be represented as follows:

$$\min_{\mathbf{w}} \left(\frac{1}{2} \|\mathbf{w}\|_2^2 + C^+ \sum_{[i|y_i=+1]} \varepsilon_i + C^- \sum_{[i|y_i=-1]} \varepsilon_i - \Phi(\rho_i) \right) \quad (7)$$

s.t. $y_i (\langle \mathbf{w}, \mathbf{P}_s^T \bar{\mathbf{x}}_s^i \rangle + b) \geq 1 - \varepsilon_i$, $\varepsilon_i \geq 0$, $\forall (\bar{\mathbf{x}}_s^i, y_i) \in \mathcal{D}_s^s$

where $\Phi(\rho_i) = \frac{1}{2} \text{Abs}(\mathbf{w}) \mathbf{r}^T$, $\text{Abs}(\mathbf{w}) = [|w_1|, |w_2|, \dots, |w_{d_v}|]$ and $\mathbf{r} = [\rho_1, \dots, \rho_{d_v}]$ is the correlation vector in which each element denotes the correlation coefficient of CCA subspace for each pair of projection dimension. Parameter ε_i is slack variable as in standard SVM. It should be noted that only labeled source data $\bar{\mathbf{x}}_s^i \in \mathcal{D}_s^s$ is available for training, and y_i is the associated class label. We put $\mathbf{P}_s^T \bar{\mathbf{x}}_s^i$ as the projection of source project data $\bar{\mathbf{x}}_s^i$ onto the correlation subspace.

In (7), the term $\Phi(\rho_i)$ is introduced for model adaptation based on CCA. By doing so, a smaller correlation coefficient ρ_i is obtained for the i^{th} dimension of CCA subspace, then the above equation would enforce the reduction of the corresponding $|w_i|$ and restrict the trained SVM model along that dimension. On the other side, a larger ρ_i favors the contribution of the associated $|w_i|$ when minimizing (7).

Since it is not forthright to solve the minimization problem in (7), we seek the approximated solution by modifying the correlation regularizer term $\Phi(\rho_i)$ into the following form:

$$\Phi(\rho_i) = \frac{1}{2} \text{Abs}(\mathbf{w} \odot \mathbf{w}) (\mathbf{r} \odot \mathbf{r})^T \quad (8)$$

where \odot denotes the element-wise multiplication. By incorporating (8) into (7), the objective function can be rewritten into a unified form:

$$\min_{\mathbf{w}} \left(\frac{1}{2} \sum_{i=1}^{d_c} (1 - \rho_i^2) w_i^2 + C^+ \sum_{[i|y_i=+1]}^N \varepsilon_i + C^- \sum_{[i|y_i=-1]}^N \varepsilon_i \right)$$

$$s.t. \quad y_i (\langle \mathbf{w}, \mathbf{P}_s^T \bar{\mathbf{x}}_s^i \rangle + b) \geq 1 - \varepsilon_i, \quad \varepsilon_i \geq 0, \quad \forall (\bar{\mathbf{x}}_s^i, y_i) \in \mathcal{D}_i^s \quad (9)$$

We refer to (9) as our cost-sensitive correlation transfer SVM. Since the correlation coefficient ρ_i ranges from 0 to 1, the above object function is a convex optimization problem. We apply the Newton-Armijo algorithm for solving SVM optimization problems. As a result, our modified SVM could adapt the derived classification model \mathbf{w} relied on the cross-project transfer ability of CCA. Moreover, we assign two misclassification costs to alleviate the effect of class imbalance. Thus, it can present the better classification performance in correlation subspace. The decision function for classifying the test modules in target project is shown as follows:

$$f(\mathbf{x}) = \text{sgn}(\langle \mathbf{w}, \mathbf{P}_t^T \bar{\mathbf{x}}_t \rangle + b) \quad (10)$$

where \mathbf{P}_t^T projects the target data $\bar{\mathbf{x}}_t$ from the target space onto the correlation subspace. Finally, $\text{sgn}(z)$ returns 1 if and only if $z > 0$, and -1 otherwise.

IV. EXPERIMENTS

A. Experimental Datasets

We collect publicly available defect datasets from prior researches, including NASA, SOFTLAB, AEEEM and ReLink [7][8]. Among these datasets, the percentage of defective components ranges from 8.65% to 50.52%. It is obvious that most datasets are imbalanced. Table I shows detailed project information in our experiments.

B. Experimental Design

To validate the effectiveness of the proposed approach for HCPDP, we compare our approach with several representative methods including TNB [5], TCA+ [6], CCA+ [8] and NN-filter [13]. We design the three experiments to evaluate our approach: (1) HCPDP with partially different metrics. We build model using common metrics between source and target datasets as in previous studies [8]. (2) HCPDP with entire different metrics. In this part, we present CCA+ and the within-project defect prediction results as references. (3) The impact of different class-imbalance rates on HCPDP, exploring whether or not our proposed method can effectively deal with class-imbalance problem in HCPDP.

For WPDP, we employ the standard SVM as their base classifier. We use the 50:50 random splits to obtain training and test sets. Thus, we repeat this process 30 times to get the average prediction results. In our approach, in order to emphasize the risk cost, the parameters C^+ and C^- are set as $C^+ : C^- = 5 : 1$. For different projects, user can select different ratios [21]. The parameter ε is determined by searching a wide range and choosing the one which produces the best F-measure value. Although we have verified that these choices of parameters work well in our experiments, we recognize that a finer tuning of them may further improve the performance.

To evaluate the performance of our method, we use two widely-used evaluation measures, F-measure and Area Under the Receiver Operating Characteristic (ROC) Curve (AUC). F-measure is the harmonic mean of *precision* and *recall*, falling

TABLE I. HETEROGENEOUS DATASETS FROM DIFFERENT PROJECTS

Group	Dataset	Instances	Buggy (%)	Metrics
NASA	CM1	327	42(12.84%)	37
	MW1	253	27(10.67%)	
	PC1	705	61(8.65%)	
AEEEM	EQ	325	129(39.69%)	61
	JDT	997	206(20.66%)	
	LC	399	64(9.26%)	
	ML	1862	245(13.16%)	
	PDE	1492	209(14.01%)	
ReLink	Apache	194	98(50.52%)	26
	Safe	56	22(39.29%)	
	ZXing	399	118(29.57%)	
SOFTLAB	AR3	63	8(12.72%)	29
	AR4	107	20(18.69%)	
	AR5	36	8(22.22%)	

in the range [0, 1], as (11). The *recall* is defined as the ratio of the number of modules correctly classified as defect to the number of defective modules. The *precision* is the ratio of the number of modules correctly classified as defect to the number of modules classified as defect.

$$F - \text{measure} = \frac{2 \times \text{recall} \times \text{precision}}{\text{recall} + \text{precision}} \quad (11)$$

For more comprehensive evaluation of predictors in the imbalanced context, the AUC is exploited to evaluate the prediction performance. AUC estimates the area under the ROC curve, which illustrates the trade-off between detection and false alarm rates, varying in [0, 1]. A better classifier should produce a higher F-measure and AUC.

C. Experimental results

Table II and Table III show that the F-measure and AUC values of heterogeneous defect prediction, where 28 common metrics exist in source and target data. In each table, the best performance are presented with boldface in all experimental datasets. The last rows of tables show the average performances on all the experimental datasets.

From Table II, we see that CCT-SVM can obtain better F-measure values in most prediction scenes, compared with other methods, and the average F-measure of CCT-SVM is the highest. This indicates that after carefully taking the class imbalance nature of defect data into consideration, CCT-SVM is able to improve defect performance dramatically. The trends in Table III are similar to that shown in Table II. The reason is that our method uses all metrics rather than only common metrics, and these metrics usually contain some useful discriminant information. We exploit the cross-project transfer ability in derived subspace when designing the associated SVM classifier. And we emphasize the risk cost to make the classification inclining to classify a module as a defective one, alleviating the impact of imbalanced data. Wilcoxon's rank sum test at a 0.05 significance level indicates that performance improvement on each pair dataset is statistical significance. This fact suggests that addressing the class imbalance problem is beneficial to construct better predictive model in software defect prediction.

In practical scenario, we often confront the situations that there are no common metrics between source and target project data. Hence we conduct experiment to investigate the performance of CCT-SVM with entire different metrics. In this section, we conduct within-project prediction results of a target project as baseline. Specifically for each dataset, we randomly

TABLE II. MEDIAN F-MEASURES WITH 28 COMMON METRICS

Source⇒Target	NN	TNB	TCA+	CCA +	CCT-SVM
CM1 ⇒ AR3	0.403	0.271	0.333	0.582	0.612
CM1 ⇒ AR4	0.632	0.337	0.416	0.772	0.756
CM1 ⇒ AR5	0.293	0.325	0.376	0.686	0.711
PC1 ⇒ AR3	0.596	0.467	0.323	0.791	0.802
PC1 ⇒ AR4	0.551	0.331	0.373	0.716	0.733
PC1 ⇒ AR5	0.512	0.379	0.516	0.723	0.719
MW1 ⇒ AR4	0.591	0.326	0.396	0.689	0.707
AR4 ⇒ CM1	0.256	0.296	0.279	0.781	0.786
AR4 ⇒ PC1	0.281	0.337	0.219	0.712	0.751
AR4 ⇒ MW1	0.523	0.386	0.433	0.768	0.781
AVG	0.463	0.346	0.366	0.722	0.736

TABLE III. MEDIAN AUCS WITH 28 COMMON METRICS

Source⇒Target	NN	TNB	TCA+	CCA +	CCT-SVM
CM1 ⇒ AR3	0.583	0.556	0.543	0.692	0.703
CM1 ⇒ AR4	0.550	0.509	0.539	0.709	0.705
CM1 ⇒ AR5	0.581	0.627	0.619	0.744	0.771
PC1 ⇒ AR3	0.601	0.593	0.639	0.751	0.763
PC1 ⇒ AR4	0.621	0.608	0.613	0.753	0.759
PC1 ⇒ AR5	0.653	0.686	0.673	0.839	0.856
MW1 ⇒ AR4	0.507	0.567	0.556	0.690	0.705
AR4 ⇒ CM1	0.501	0.530	0.522	0.694	0.691
AR4 ⇒ PC1	0.431	0.489	0.453	0.571	0.584
AR4 ⇒ MW1	0.473	0.516	0.513	0.627	0.631
AVG	0.550	0.570	0.567	0.707	0.721

choose the 50% samples as the training data and the other 50% are testing data. We repeat this process 30 times and report the average prediction results.

Table IV and V show the F-measure and AUC of different compared methods, where no common metrics exist in the source and target data. From Table IV and Table V, we can see that CCT-SVM can obtain better results in contrast with the CCA+ and within-project prediction in most cases. The results suggest that our method takes the misclassification costs into consideration, which makes the prediction tending to classify the defective-free modules as the defective ones in order to get higher prediction performance. Table V tabulates the AUC values. The trends of AUC values in Table V are similar to that of F-measure shown in Table IV. Therefore, CCT-SVM can be used to address HCPDP effectively.

CCT-SVM can effectively address heterogeneous defect prediction problem even if the class distribution is imbalanced. In order to study the influence of the different class-imbalance rates on CCT-SVM under heterogeneous cross-project setting, we conduct additional experiments, where we alter the different classes distribution of the source data which is customized so that the number of the defective samples over the number of the defective-free samples is roughly δ , $1/\delta \in \{1, 2, \dots, 10\}$. If the original proportion is larger than δ , we randomly abandon some defective samples; otherwise, we randomly abandon some defective-free samples. Here, we build a prediction model by using the customized source project data and then apply the model to the target project data.

We repeat the experiment in each customized dataset for 30 times. We plot the average F-measures and AUCs versus the inverse of the minority-majority ratio ($1/\delta$) on the experimental datasets. We only report the experimental results on the three pair representative datasets: MW1 ⇒ AR4 (28 common metrics), ZXing ⇒ AR4 (3 common metrics), JDT ⇒ ZXing (no common metrics), as shown in Fig. 2-4.

TABLE IV. MEDIAN F-MEASURES WITH NO COMMON METRICS

Source⇒Target	CCA +	CCT-SVM	Within Target⇒Target
CM1 ⇒ EQ	0.581	0.612	0.576
EQ ⇒ CM1	0.238	0.276	0.336
LC ⇒ Apache	0.266	0.307	0.645
Apache ⇒ LC	0.288	0.291	0.373
ML ⇒ PC1	0.541	0.567	0.369
JDT ⇒ PC1	0.501	0.523	
PDE ⇒ PC1	0.431	0.455	
ML ⇒ AR4	0.573	0.559	0.392
JDT ⇒ AR4	0.493	0.517	
PDE ⇒ AR4	0.540	0.536	
PC1 ⇒ ML	0.336	0.333	0.273
AR4 ⇒ ML	0.353	0.336	
ZXing ⇒ ML	0.405	0.446	
PC1 ⇒ JDT	0.521	0.533	0.563
AR4 ⇒ JDT	0.592	0.631	
ZXing ⇒ JDT	0.647	0.675	
PC1 ⇒ PDE	0.376	0.401	0.312
AR4 ⇒ PDE	0.383	0.413	
ZXing ⇒ PDE	0.421	0.473	
ML ⇒ ZXing	0.486	0.501	0.336
JDT ⇒ ZXing	0.466	0.508	
PDE ⇒ ZXing	0.474	0.497	
AVG	0.451	0.475	0.394

TABLE V. MEDIAN AUCS WITH NO COMMON METRICS

Source⇒Target	CCA +	CCT-SVM	Within Target⇒Target
CM1 ⇒ EQ	0.711	0.752	0.651
EQ ⇒ CM1	0.798	0.816	0.728
LC ⇒ Apache	0.806	0.797	0.769
Apache ⇒ LC	0.718	0.757	0.608
ML ⇒ PC1	0.861	0.827	0.796
JDT ⇒ PC1	0.759	0.823	
PDE ⇒ PC1	0.791	0.815	
ML ⇒ AR4	0.765	0.809	0.654
JDT ⇒ AR4	0.676	0.717	
PDE ⇒ AR4	0.730	0.766	
PC1 ⇒ ML	0.673	0.726	0.754
AR4 ⇒ ML	0.653	0.696	
ZXing ⇒ ML	0.725	0.746	
PC1 ⇒ JDT	0.720	0.751	0.809
AR4 ⇒ JDT	0.612	0.679	
ZXing ⇒ JDT	0.751	0.883	
PC1 ⇒ PDE	0.702	0.727	0.711
AR4 ⇒ PDE	0.681	0.719	
ZXing ⇒ PDE	0.731	0.701	
ML ⇒ ZXing	0.684	0.652	0.609
JDT ⇒ ZXing	0.667	0.723	
PDE ⇒ ZXing	0.721	0.707	
AVG	0.724	0.751	0.715

As expected, F-measure and AUC values of all the compared methods decrease as the dataset becomes more imbalanced, but the influence of the increase of class imbalance on CCT-SVM is the smallest. Fig. 2-4 show that CCT-SVM almost always performs better than the other methods, and when the class distribution is more imbalanced the superiority is more preponderant. This fact suggests that the degree of imbalance has great influence on HCPDP, if it does not address the class imbalance problem explicitly. Therefore, it can be concluded that explicitly tackling the class-imbalance problem is helpful to HCPDP.

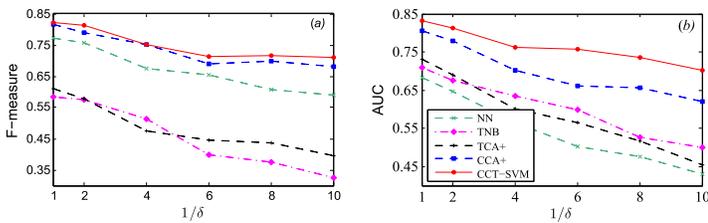


Figure 2. The performance of compared methods on MW1⇒AR4 (28 common metrics) at different minority-majority.

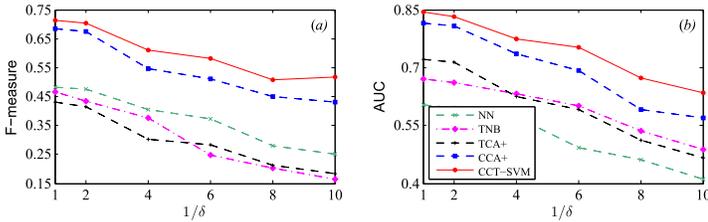


Figure 3. The performance of compared methods on ZXing⇒AR4 (3 common metrics) at different minority-majority

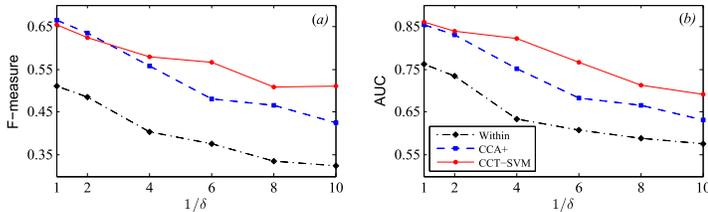


Figure 4. The performance of compared methods on JDT⇒ZXing (no common metrics) at different minority-majority

V. CONCLUSION AND FUTURE WORK

Cross-project software defect prediction plays an important role in improving the quality of a software product in case of projects without sufficient historical data. However, it is difficult to conduct with heterogeneous metrics set. In addition, software defect datasets have the class-imbalance characteristic. Without taking this issue into account, the effectiveness of software defect prediction would be greatly reducing. In this paper, we addressed these two important issues simultaneously and proposed a novel cost-sensitive correlation transfer support vector machine method for heterogeneous defect prediction. Experimental results on the open source projects from different groups showed that our method is feasible and yields promising results.

For the future work, we will introduce other sophisticated class imbalance learning techniques in the heterogeneous cross-project defect prediction, and we will evaluate our approach in more heterogeneous defect datasets.

ACKNOWLEDGMENT

The research in this paper was partially supported by the National Natural Science Foundation of China under Projects No. 61373039, No. 61170022, No. 61003071 and No. 91118003).

REFERENCES

[1] G. Czibula, Z. Marian, and I. G. Czibula, "Software defect prediction using relational association rule mining," *Information Sciences*, vol. 264, pp. 260–278, 2014.

[2] X. Y. Jing, S. Ying, Z. W. Zhang, S. S. Wu, and J. Liu, "Dictionary learning based software defect prediction," In *Proceedings of the 36th International Conference on Software Engineering*, 2014, pp. 414–423.

[3] T. Menzies, J. Greenwald, and A. Frank, "Data mining static code attributes to learn defect predictors," *Software Engineering, IEEE Transactions on*, vol. 33, no.1, pp. 2–13, 2007.

[4] I. H. Laradji, M. Alshayeb, and L. Ghouti. Software defect prediction using ensemble learning on selected features. *Information and Software Technology*, 58:388–402, 2015.

[5] Y. Ma, G. C. Luo, X. Zeng and A. Chen, "Transfer learning for crosscompany software defect prediction," *Information and Software Technology*, vol. 54, no. 3, pp. 248–256, 2012.

[6] J. Nam, S. J. Pan and S. Kim, "Transfer defect learning," *Proceedings of the 35th International Conference on Software Engineering*. San Francisco, 2013, pp. 382–391.

[7] J. Nam and S. Kim, "Heterogeneous defect prediction," In *Proceedings of the 10th Joint Meeting on Foundations of Software Engineering*, 2015, pp. 508–519.

[8] X. Y. Jing, F. Wu, X. Dong, F. Qi, and B. Xu, "Heterogeneous cross-company defect prediction by unified metric representation and cca-based transfer learning," In *Proceedings of the 10th Joint Meeting on Foundations of Software Engineering*, 2015, pp. 496–507.

[9] Y. Jiang, M. Li, and Z.-H. Zhou, "Software defect detection with rocus," *Journal of Computer Science and Technology*, vol. 26, no. 2, pp. 328–342, 2011.

[10] D. Ryu, O. Choi, and J. Baik, "Value-cognitive boosting with a support vector machine for cross-project defect prediction," *Empirical Software Engineering*, vol. 21, no. 1, pp. 43–71, 2016.

[11] T. Zimmermann, N. Nagappan, H. Gall, E. Giger, and B. Murphy, "Cross-project defect prediction: a large scale experiment on data vs. domain vs. process," In *Proceedings of the 7th joint meeting of the European software engineering conference and the ACM SIGSOFT symposium on The foundations of software engineering*, 2009, pp. 91–100.

[12] Z. He, F. Shu, Y. Yang, M. Li, and Q. Wang, "An investigation on the feasibility of cross-project defect prediction," *Automated Software Engineering*, vol. 19, no. 2, pp. 167–199, 2012.

[13] B. Turhan, T. Menzies, A. B. Bener, and J. Di Stefano, "On the relative value of cross-company and within-company data for defect prediction," *Empirical Software Engineering*, vol. 14, no. 5, pp. 540–578, 2009.

[14] T. Hall, S. Beecham, D. Bowes, D. Gray, and S. Counsell, "A systematic literature review on fault prediction performance in software engineering," *Software Engineering, IEEE Transactions on*, vol. 38, no. 6, pp. 1276–1304, 2012.

[15] S. Wang and X. Yao, "Using class imbalance learning for software defect prediction," *Reliability, IEEE Transactions on*, vol. 62, no. 2, pp. 434–443, 2013.

[16] J. Zheng, "Cost-sensitive boosting neural networks for software defect prediction," *Expert Systems with Applications*, vol. 37, no. 6, pp. 4537–4543, 2010.

[17] T. G. Grbac, G. Mause, and B. D. Basic, "Stability of software defect prediction in relation to levels of data imbalance," In *SQAMIA*, 2013, pp. 1–10.

[18] J. Ren, K. Qin, Y. Ma, and G. Luo, "On software defect prediction using machine learning," *Journal of Applied Mathematics*, 2014.

[19] Z. Sun, Q. Song, and X. Zhu, "Using coding-based ensemble learning to improve software defect prediction," *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, vol.42, no. 6, pp. 1806–1817, 2012.

[20] Y. R. Yeh, C. H. Huang, and Y. C. F. Wang, "Heterogeneous domain adaptation and classification by exploiting the correlation subspace," *Image Processing, IEEE Transactions on*, vol. 23, no. 5, pp. 2009–2018, 2014.

[21] Y. Jiang, B. Cukic, and T. Menzies, "Cost curve evaluation of fault prediction models," In *Software Reliability Engineering, ISSRE, 19th International Symposium on*, pp. 197–206, 2008.

Long-Term Active Integrator Prediction in the Evaluation of Code Contributions

Jing Jiang, Fuli Feng, Xiaoli Lian, Li Zhang

State Key Laboratory of Software Development Environment, Beihang University, Beijing, China
{jiangjing, lily}@buaa.pku.edu.cn, fulifeng93@gmail.com, lxl_tuizi@hotmail.com

Abstract—In open source software (OSS) projects, integrators are given high-level access to repositories so that they could maintain and manage projects. Although integrators play a critical role in evaluating code changes for OSS projects, they may be short-term active. Long-term active integrators keep in evaluating code update submission and managing responses from contributors. In order to survive and succeed, OSS projects need to attract and retain long-term active integrators. To assist OSS projects to retain active integrators, we propose a method called LTAPredict to predict whether integrators will be long-term active in the evaluation of code contributions. LTAPredict collects activity data of integrators, extracts a rich set of features, and makes prediction via machine learning techniques. We perform experiments on 37 popular projects, containing a total of 1,073 integrators. Results show that based on the Decision Tree, LTAPredict achieves the accuracy as 0.829, the precision as 0.81, the recall as 0.827 and the F1 as 0.818. Meanwhile, we evaluate the feature importance to identify the most significant indicators of long-term active integrators. We observe that whether integrators becoming long-term active is associated with the number of active months and social distance with contributors in their first year as integrators. These findings assist OSS projects to identify potential long-term active integrators and adopt better strategies to retain them in the evaluation of code contributions.

Keywords—Long-term active integrator; Code contributions; Open source software

I. INTRODUCTION

In open source software (OSS) projects, contributors fix bugs or improve functions, and then submit these code changes to original projects [1]. Integrators are granted the privilege of directly committing codes to source code repositories [2], [3], [4]. Integrators are responsible for reviewing code updates submitted by contributors, and deciding whether or not merge these code changes into repositories. Integrators not only merge satisfactory codes into repositories, but also discuss with contributors and encourage them to keep in submitting codes.

Although integrators play a critical role in evaluating code changes for OSS projects, they may be short-term active in the evaluation of code contributions. Due to the principle of voluntary participation, integrators always have the freedom to decide their activities and even leave the community [5]. Furthermore, some integrators may focus on writing codes by themselves, and seldom spend time in evaluating code contributions from others. Long-term active integrators keep in

evaluating code modifications and merging satisfactory codes into repositories. Long-term active integrators in the evaluation are crucial for projects success, because their participation in a long period keeps the sustainable development for OSS projects. In addition, if integrators always change, it is hard for contributors to know evaluative criteria, which may further discourage volunteers to make contributions [6].

Previous works have explored how new participants join projects [1], make contributions [7], [8], enter the circle of trust [9] and finally become integrators [10]. Some other works have studied factors which impact decisions in the evaluation of code contributions [2], [3]. However, it remains unknown whether integrators are active in a long time or not, and what affect their decisions. Thus there is a need for a comprehensive analysis of long-term active integrators in the management of code contributions.

In this work, we explore what make long-term active integrators in open source project-hosting site GitHub [11]. GitHub is one of the world's largest open source communities, and it provides the pull request model for contributors to submit code updates [2]. We mainly study pull requests to understand integrators' contributions in code evaluation. We use GitHub API to collect data, and obtain information of 37 popular projects and their 1,073 integrators. We study active periods of integrators in the management of code contributions. Based on analysis results, we define integrators as long-term active if they have joined projects for more than 1 year and keep active in evaluating code contributions in more than 30% of months.

We propose a method called LTAPredict to predict whether integrators will be long-term active. LTAPredict extracts various kinds of features to comprehensively describe integrators, including willingness and capacity, environment, experience and social status. LTAPredict uses these features and makes prediction via machine learning techniques. Experimental results show that based on the Decision Tree, LTAPredict achieves the accuracy as 0.829, the precision as 0.81, the recall as 0.827 and the F1 as 0.818. Meanwhile, we evaluate the feature importance to identify the most significant indicators of long-term active integrators. We observe that whether integrators becoming long-term active is associated with the number of active months and social distance with contributors in their first year as integrators. These findings assist OSS projects to identify potential long-term active integrators and adopt better strategies to retain them in the evaluation of code

contributions.

II. BACKGROUND AND DATASETS

In this section, we begin by providing background information about contribution evaluation process in GitHub. Then, we introduce how our datasets are collected. Finally, we report statistics of our datasets.

A. Contribution Evaluation Process

In GitHub, contributors fork repositories, use codes as their own and make changes [2]. Contributors submit pull requests when they want to merge their changes into main repositories. Integrators inspect code changes and evaluate potential contributions. Integrators have several options towards pull requests: If pull requests are deemed satisfactory, they merge code changes into main repositories; otherwise, they may directly reject and close them, or ask contributors to make updates. Integrators may also ignore code contributions, and leave pull requests open. For each pull request, an issue is opened automatically, where integrators and other interested users exchange comments.

Integrators, who evaluate the quality of code change submission, decide project evolution and play a critical role for OSS projects. In order to becoming integrators, developers need to successfully pass the evaluation of technical contributions and social interactions [7], [9], [10]. After becoming integrators, they are given high-level access to repositories, and have the responsibility of evaluating code update submission and managing responses from contributors. Although integrators play a critical role in evaluating code changes for OSS projects, they may be short-term active in the evaluation of code contributions. The principle of voluntary participation attracts a lot of people to join the OSS community, while it also means that people always have the freedom to decide their activities or even leave the community [5]. Furthermore, some integrators may focus on writing codes by themselves, and seldom spend time in evaluating code contributions from others.

B. Data Collection

GitHub provides access to its internal data stores through an API¹. It allows us to access a rich collection of OSS projects, and provides valuable opportunities for research. We gather information from GitHub API and create a dataset of projects and integrators. The process of data collection is as follow:

We obtain project lists from MSR 2014 Mining Challenge MySQL Dataset [12]. This dataset includes 90 popular software projects for top programming languages in GitHub. We focus on popular and active projects, because they may need integrators. Small projects often have few integrators, and their project owners can manage projects by themselves.

Next, we downloaded historical information of 90 projects in July 2014, including their pull requests. Then the following criteria is applied to exclude projects from the initial selection: Firstly, projects should have at least one event of activities

within 1 month prior to data collection (July 2014), so as to avoid inactive projects. Secondly, projects should be created at least two years prior to data collection. It ensures that projects have more than two years of historical information. We are interested to explore how their integrators behave as time goes on. Finally, projects should have at least 300 pull requests. For our analysis, we use the evaluation process of pull requests to identify active integrators. We choose projects which use pull requests as the important method for code contributions in GitHub.

After selection, our sample includes 37 projects. We make basic analysis of projects in our dataset. 8 projects were created earlier than July 2009, and have histories longer than 5 years; 20 projects have histories longer than 4 years, and 32 projects have histories longer than 3 years. Since GitHub was founded in April 2008², these projects have long histories in GitHub and provide great opportunities to explore evolution of integrators. Our dataset includes projects written in representative languages, such as PHP, Ruby, Python, C, C++, JavaScript and Scala [13].

We collect detailed information of pull requests from all developers in 37 projects through GitHub API. Then we collect project integrators from pull requests. GitHub does not provide the exact promotion time of integrator, and we use the first activity time to estimate the promotion time and determine the integrator. If a developer firstly merges the pull request, or closes a pull request from another developer, then this developer is considered as an integrator, and the promotion time is estimated by this first activity time. We exclude activity records before the first activity time, so as to make sure that the user has already been promoted as the integrator. Next, we extract pull request records for integrators. Integrators take different kinds of actions to handle pull requests, such as accepting, rejecting, closing or discussing pull requests. We extract these activities from our datasets, and build records about *when* pull requests are handled by *which* integrators with *which* actions. GitHub is a social coding site [11], and allows users to attract followers. We also collect follower and following relationships of integrators. Finally, our dataset includes 1,073 integrators and their 104,910 records of pull request evaluation.

C. Basic Statistics

In this subsection, we report basic statistics of our datasets. The integrator's age is defined as the number of months between the integrator's promotion time and data collection. For every active integrator, we compute the number of active months after the promotion, divided by the integrator's age. In our computation, we exclude integrators who join projects less than 1 year before our data collection. This is because that the time is too short to study their characteristics. Figure 1 shows complementary cumulative distribution function (CCDF) of the percentage of active months for active integrators. 64.1% of integrators have the percentage of active months larger

¹<http://developer.github.com/v3/>

²<https://github.com/blog/40-we-launched>

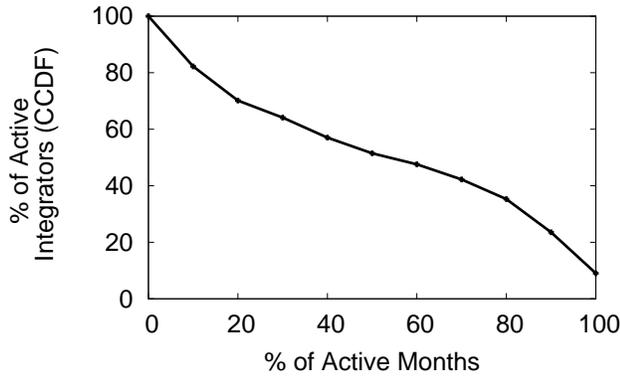


Fig. 1. Percentage of active months

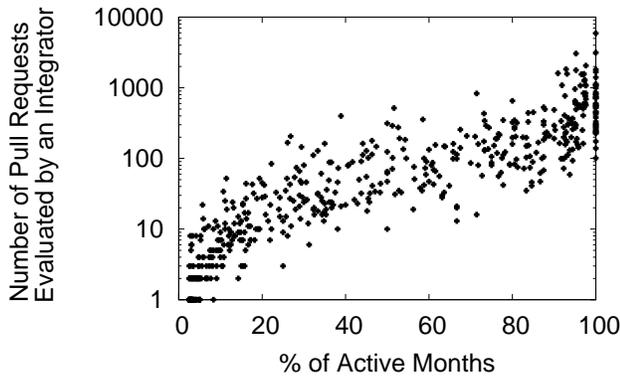


Fig. 2. The relationship between active periods and productivity

than 30%, while other 35.9% of active integrators have the percentage of active months less than 30%. The majority of active integrators keep in evaluating code contributions in a long time, while a minority of integrators occasionally handle pull requests.

We study the relationship between the number of active months and the amount of pull requests evaluated by the integrator. Figure 2 shows the percentage of active months versus the number of pull requests evaluated by the integrator. When an integrator has the larger percentage of active months, this integrator is likely to evaluate more pull requests. The Pearson correlation between the percentage of active months and the number of pull requests is as high as 0.51. The productivity in contribution evaluation is strongly correlated with the active time.

We consider the definition of long-term active and short-term active integrators in the evaluation of pull requests. In fact, there is no standard for the definition, and different people may have various attitude. The definition of long-term active integrators is mainly used to study impact factors of their activities. We also try other thresholds of the percentage of active months, and observe similar results of feature importance and prediction results. Figure 1 shows that 64.1% of active integrators have the percentage of active months larger than 30%. We choose 30% as the threshold to classify active integrators. According to active periods and productivity, we classify active integrators into three categories: *short-term*

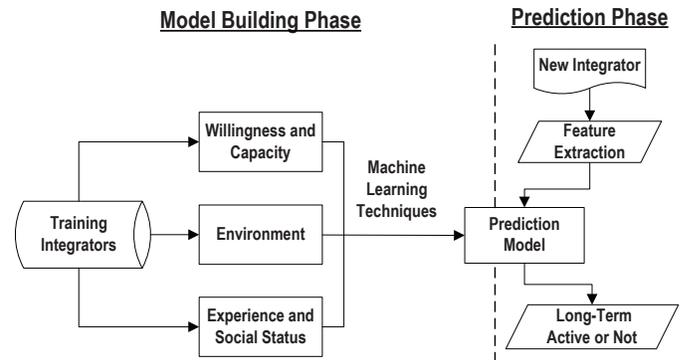


Fig. 3. Overall framework of our method LTAPredict

active, long-term active and unclassified. Firstly, integrators are *long-term active* if they have joined projects for more than 1 year, and keep active in evaluating code contributions in more than 30% of months. Joining projects at least 1 year before data collection ensures that our data is enough for statistical analysis. Secondly, integrators are *short-term active* if they have joined projects for more than 1 year, but keep active in evaluating code contributions in less than 30% of months. Finally, if integrators join projects less than 1 year before our data collection, there is not enough data to make analysis. These integrators are *unclassified*, and they are excluded in our research and left for future research. Figure 2 shows that the productivity has strong correlation with active periods. Therefore, the percentage of active months is enough to measure the productivity of integrators.

III. LTAPREDICT: A METHOD TO PREDICT LONG-TERM ACTIVE INTEGRATORS

In this section, we describe our method LTAPredict to predict long-term active integrators. Figure 3 presents the overall framework of LTAPredict. The entire framework contains two phases: a model building phase and a prediction phase. In the model building phase, our goal is to build a model from training datasets. In the prediction phase, the model is used to predict whether an integrator will be long-term or short-term active.

As described in the subsection II-B, activity data of integrators is firstly collected. Next, we generate features for our analysis based on prior literature about pull request acceptance [2], [3], long-term contributor [1], social coding [11], [14] and committer promotion [9], [10]. We split selected features into three categories, including willingness and capacity, environment, experience and social status. We make basic statistics of these features in Table I. We describe detailed definitions and why we choose these features in subsections III-A, III-B and III-C. Then we use different machine learning techniques to build the prediction model, as described in the subsection III-D. In the prediction phase, we use the LTAPredict to predict whether an integrator will be long-term or short-term active.

TABLE I
SELECTED FEATURES AND BASIC STATISTICS OF ACTIVE INTEGRATORS

Feature	5%	Mean	95%
Willingness and Capacity			
initial_active_month	12	7.19	1
handle_time (day)	49.83	13.8	0.28
num_comments	15	6.35	1
Environment			
total_pulls	2,872	707.95	17
Experience and Social Status			
age (day)	1448.83	728.71	91.23
social_distance	0.42	0.09	0
num_follower	1680	326.77	1
num_following	76	17.61	1

A. Willingness and Capacity

The probability for a new developer to become a long-term contributor may be influenced by the willingness and capacity [15]. Willing and capacity are important factors to decide contributors' activeness in future, and they may also influence integrators' activities. We use three features to measure integrators' willingness and capacity.

Initial Active Month: Gharehyazie et al. found that the number of patches a contributor submitted in early months was the important factor to predict the promotion of integrators [10]. We wonder whether activity frequency is also important after developers become integrators. We consider several features to describe activity frequency of integrator in early months: the number of pull requests the integrator evaluates in the first year as an integrator, the number of pull requests the integrator evaluates divided by the total number of pull requests in the first year, and the number of active months in the first year. To check whether these features are sufficiently independent, we leverage Spearman's rank correlation coefficient [16] (ρ). We observe that ρ between any two features is more than 0.8, and these features are highly correlated. Therefore, our analysis uses the number of active months in the first year as an integrator.

Handle Time: After a contributor submits the pull request, it takes some time for an integrator to evaluate the contribution and give feedback. We compute the average interval time between the pull request's submission and the integrator's first responding. It describes whether the integrator handles pull requests quickly or slowly. Note that the integrator may take several actions towards a pull request, such as leaving comments and then closing the pull request. We use the first action in this feature. The interval time between the pull request's submission and the integrator's last responding is not used, because it is highly correlated with that of first responding.

Num Comments: Marlow et al. observed that uncertain contributions required explanation and discussion [17]. Pull requests with many of comments may be more complicated to evaluate [3], and cost integrators additional time to negotiate with submitters. Due to the high degree of uncertainty, integrators may be unwilling to evaluate these contributions. To measure the level of discussion, we compute the average number of comments in pull requests handled by the integrator.

B. Environment

Previous work showed that the project environment at the joining time had obvious impact on the contributors' behavior [15]. We explore whether project environment influences integrators' activities.

Total Pulls: To study environment impact on the integrator, we count the total number of pull requests accumulated in the project, when the developer becomes an integrator. This feature has strong correlation with other measures, including the project's age [10], current size of core team (number of active integrators) [2], [3] and number of contributors who have submitted pull requests before. Therefore, the total number of pull requests is a representative feature to describe the project environment.

C. Experience and Social Status

Sinha et al. observed that the prior experience influenced the probability of the promotion from contributors to integrators [9]. Tsay et al. found that the contributors' social status affected the probability of pull request acceptance [3]. We wonder whether experience and social status are also associated with the likelihood of integrators becoming long-term active. We use four features to measure the experience and social status.

Age: We measure the age of the integrator when he or she firstly handles the pull request in the project. It is calculated as the interval time between the integrator's registration in GitHub and the first pull request evaluation in this project. It measures whether the integrator is new or old in GitHub.

Social Distance: GitHub is a social coding site, and integrates social media functionality with code management tools [11]. Users follow interesting developers and build social connections. Tsay et al. found that pull request acceptance was influenced by social connections between integrators and submitters [3]. Social connections may also influence activeness of integrators. For pull requests handled by the integrator, we compute the percentage of submitters who directly follow this integrator.

Num Follower: This feature is the number of followers the integrator has at the data collection time. The number of followers shows the status of the integrator in the OSS community [3], [14]. A large number of followers means that the integrator is more attractive and influential. We observe the number of followers is highly correlated with the number of repositories owned by the integrator. The number of follower also indicates how actively the integrator creates projects.

Num following: This feature is the number of developers followed by the integrator. It describes whether the integrator actively builds social connections with others.

D. Prediction

We use machine learning techniques to make prediction. In training datasets, we know whether the integrator becomes long-term active or not. Each integrator belongs to a category, namely long-term active or short-term active. Several machine learning algorithms are known to perform well and have been

TABLE II
MEAN PERFORMANCE ON 10-FOLD CROSS VALIDATION

Model	AUC	ACC	PRE	REC	F1
Decision Trees	0.876	0.829	0.81	0.827	0.818
Logistic Regression	0.899	0.808	0.846	0.717	0.776
Neural Network	0.882	0.792	0.796	0.743	0.769
Random Forests	0.848	0.82	0.789	0.835	0.811
Support Vector Machines	0.862	0.817	0.817	0.772	0.794
k-Nearest Neighbor	0.752	0.728	0.795	0.557	0.655

used in previous works [2], [18]. We run training datasets and build the prediction model of LTAPredict through machine learning classifiers, including Decision Trees (DT), Logistic Regression (LR), Neural Network (NN), Random Forests (R-F), Support Vector Machines (SVM) and k-Nearest Neighbor (kNN). We implement LTAPredict on top of the software Rapidminer Toolkit³. According to project requirements and experimental results, a suitable machine learning classifier can be chosen for LTAPredict.

For a new integrator, we use LTAPredict to predict the category which the integrator will belong to. If the category is long-term active, this integrator is predicted to be long-term active in the evaluation of code contributions; otherwise, the integrator is predicted to be short-term active.

IV. EXPERIMENTS AND RESULTS

In this section, we evaluate our method LTAPredict. The experimental environment is a windows server 2012, 64-bit, Intel(R) Xeon(R) 1.90 GHz server with 24GB RAM. We consider 5 metrics to evaluate experimental results, including Precision (PRE), Recall (REC), F-measure (F1), Accuracy (ACC) and Area under the receiver operating characteristic curve (AUC). We use these metrics, because they have been used in the acceptance prediction of code contributions [2]. In addition, 10-fold cross validation is applied on our experiments and every value of measurement on performance is average of that of each iteration.

Table II illustrates the mean performance of LTAPredict based on different classifiers. *Based on the Decision Tree, LTAPredict achieves the accuracy as 0.829, the precision as 0.81, the recall as 0.827 and the F1 as 0.818.* Except the k-Nearest Neighbor, LTAPredict based on other classifiers gets the AUC over 0.8, and LTAPredict based on the Logistic Regression achieves the AUC as 0.899. LTAPredict based on k-Nearest Neighbor also achieves the lowest value of ACC. Based on different classifiers, LTAPredict maintains PRE at about 0.8 with a very small fluctuation. When REC is considered, LTAPredict based on the Random Forests performs the best. LTAPredict based on the Decision Trees has the F1 as 0.818, and LTAPredict based on the Random Forests has the F1 as 0.811, which are better than other classifiers. Different machine learning classifiers bring various results for LTAPredict. In practice, a specific machine learning classifier can be chosen, according to project requirements and experimental results.

Previous work [2] uses performance decrease to identify important indicators of pull request acceptance. We also use

³<http://rapidminer.com/>

TABLE III
PERFORMANCE DECREASE ON 10-FOLD CROSS VALIDATION, WHEN EACH ATTRIBUTE IS FILTERED OUT

Attribute	AUC	ACC	PRE	REC	F1
social_distance	-0.012	-0.165	-0.049	-0.439	-0.295
initial_active_month	-0.105	-0.2	-0.142	-0.435	-0.317
num_follower	-0.023	-0.127	-0.032	-0.338	-0.211
age	-0.046	-0.106	-0.026	-0.278	-0.167
num_comments	-0.027	-0.118	-0.057	-0.27	-0.174
num_following	-0.028	-0.09	-0.024	-0.232	-0.137
handle_time	-0.017	-0.051	-0.14	-0.148	-0.144
total_pulls	-0.02	-0.047	0.006	-0.148	-0.074

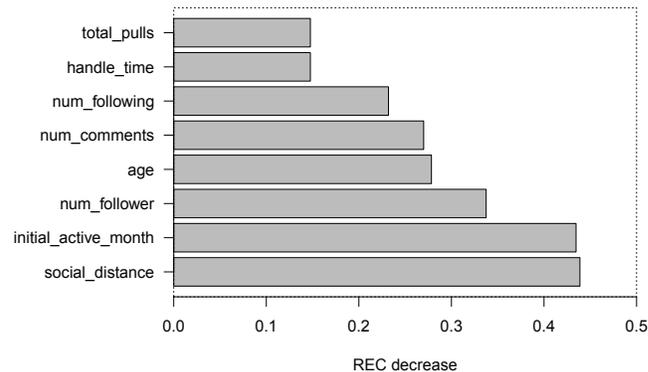


Fig. 4. Feature importance

performance decrease to estimate the importance of a feature, when it is filtered out by the select attribute operator. Since we care most on the percentage of real long-term active integrators identified by the classifier, we choose LTAPredict based on the Random Forests, which achieves the highest REC. Then we use the REC decrease to rank feature importance.

Based on the Random Forests, the LTAPredict's performance decrease caused by absence of each attribute is illustrated in Table III. It shows that REC will drop significantly if any feature is absent and some of them would bring disastrous influence on performance of LTAPredict. Rank of feature importance can be seen in Figure 4. It is obvious that an almost half drop of REC could happen without either social_distance or initial_active_month. *Therefore, whether an integrator will keep handling pull-requests is mostly decided by the social distance with the contributors and the number of active months in the first year.* In Figure 4, social_distance, number_follower and age cause great REC decrease, which shows that the experience and social distance are strongly associated with the likelihood of integrators becoming long-term active.

V. THREATS TO VALIDITY

Threats to internal validity relate to experimenter bias and errors. Firstly, there is not any standard definition of long-term active integrators. As a result, we define integrators as long-term active, if they have joined projects for more than 1 year and keep active in evaluating code contributions in more than 30% of months. Then we use this definition to identify long-term active integrators and find dominant features. We also try other thresholds of the percentage of active months and obtain similar results of feature importance and prediction

results. The choice of the percentage of active months has few impacts on our results. Secondly, we use integrators' activities in the first year to make prediction. In future work, we will explore whether integrators' activities in a shorter period can be used to make prediction. Thirdly, we use the first activity time of contribution evaluation to estimate the promotion time. If the integrator keeps inactive for a long time after the promotion, the actual promotion time is much earlier than the estimation time. In this case, the estimation time is not correct, but it has few impacts on results. This is because we mainly study activities in code evaluation, and this integrator has no activities of code evaluation between the actual promotion time and estimation time.

Threats to external validity relates to the generalizability of our study. Our empirical findings are based on open source projects in GitHub, and it is unknown whether our results can be generalized to other OSS platforms. In the future, we plan to study a similar set of research questions from other platforms such as Bitbucket, and compare their results with our findings in GitHub.

VI. RELATED WORKS

Some previous works explored how new participants joined projects, made contributions and became core members finally [1], [7], [8], [9], [10], [19], [20]. Herraiz et al. found that volunteers tended to follow a step-by-step joining process, while hired developers were usually promoted as integrators suddenly [7]. Gharehyazie et al. observed that the amount of two-way communications a person participated in, was a significant predictor of one's likelihood to becoming an integrator [10]. Our work differs in that we focus on understanding of long-term active integrators who keep in handling pull requests.

Some significant works [2], [3], [4] studied factors which affected decisions to merge pull requests in GitHub. Tsay et al. found that core members both used technical and social information to evaluate potential contributions [3]. Gousios et al. investigated factors that affected the decision to merge pull requests, and factors that affected the time it took to process pull requests [2]. We also make research on pull requests, but we mainly study integrators who evaluate these pull requests.

VII. CONCLUSION

In this work we examine what make long-term active integrators in the evaluation of code contributions. We conduct the statistical analysis of integrator activeness in GitHub. Then we use machine learning techniques to predict long-term active integrators and discover important indicators. Results show that based on the Decision Tree, LTAPredict achieves the accuracy as 0.829, the precision as 0.81, the recall as 0.827 and the F1 as 0.818. We observe that whether integrators becoming long-term active is associated with the number of active months and social distance with contributors in their first year as integrators. These findings assist OSS projects to identify potential long-term active integrators and adopt

better strategies to retain them in the evaluation of code contributions.

ACKNOWLEDGMENT

This work is supported by National Natural Science Foundation of China under Grant No.61300006, the State Key Laboratory of Software Development Environment under Grant No.SKLSDE-2015ZX-24, and Beijing Natural Science Foundation under Grant No.4163074.

REFERENCES

- [1] M. Zhou and A. Mockus, "Does the initial environment impact the future of developers?" in *Proc. of ICSE*, Honolulu, USA, May 2011.
- [2] G. Gousios, M. Pinzger, and A. van Deursen, "An exploratory study of the pull-based software development model," in *Proc. of ICSE*, Hyderabad, India, July 2014.
- [3] J. Tsay, L. Dabbish, and J. Herbsleb, "Influence of social and technical factors for evaluating contribution in github," in *Proc. of ICSE*, Hyderabad, India, July 2014.
- [4] G. Gousios, A. Zaidman, M.-A. Storey, and A. van Deursen, "Work practices and challenges in pull-based development: The integrators perspective," in *Proc. of ICSE*, Florence, Italy, May 2015.
- [5] K. Crowston, K. Wei, J. Howison, and A. Wiggins, "Free/libre open source software development: What we know and what we do not know," *ACM Computing Surveys*, vol. 44, no. 2, pp. 1–35, 2012.
- [6] L. Dabbish, C. Stuart, J. Tsay, and J. Herbsleb, "Leveraging transparency," *IEEE Software*, vol. 30, no. 1, pp. 37–43, 2013.
- [7] I. Herraiz, G. Robles, J. J. Amor, T. Romera, and J. M. G. Barahona, "The processes of joining in global distributed software projects," in *Proc. of GSD*, Shanghai, China, May 2006.
- [8] C. Jensen and W. Scacchi, "Role migration and advancement processes in ossd projects: A comparative case study," in *Proc. of ICSE*, Minnesota, USA, May 2007.
- [9] V. S. Sinha, S. Mani, and S. Sinha, "Entering the circle of trust: Developer initiation as committers in open-source projects," in *Proc. of MSR*, Honolulu, USA, May 2011.
- [10] M. Gharehyazie, D. Posnett, and V. Filkov, "Social activities rival patch submission for prediction of developer initiation in oss projects," in *Proc. of ICSM*, Eindhoven, The Netherlands, September 2013.
- [11] L. Dabbish, C. Stuart, and J. Herbsleb, "Social coding in github: Transparency and collaboration in an open software repository," in *Proc. of CSCW*, Washington, USA, February 2012.
- [12] G. Gousios, "The ghtorrent dataset and tool suite," in *Proc. of MSR*, Hyderabad, India, May 2014.
- [13] T. F. Bissyande, F. Thung, D. Lo, L. Jiang, and L. Reveillere, "Popularity, interoperability, and impact of programming languages in 100,000 open source projects," in *Proc. of COMPSAC*, Kyoto, Japan, July 2013.
- [14] J. Jiang, L. Zhang, and L. Li, "Understanding project dissemination on a social coding site," in *Proc. of WCRE*, Koblenz, Germany, October 2013.
- [15] M. Zhou and A. Mockus, "What make long term contributors: willingness and opportunity in oss community," in *Proc. of ICSE*, Zurich, Switzerland, June 2012.
- [16] E. L. Lehmann and H. J. M. D'Abrera, *Nonparametrics: Statistical Methods Based on Ranks*. Prentice-Hall, 1998.
- [17] J. Marlow, L. Dabbish, and J. Herbsleb, "Impression formation in online peer production: Activity traces and personal profiles in github," in *Proc. of CSCW*, San Antonio, USA, February 2013.
- [18] X. Xia, D. Lo, X. Wang, X. Yang, S. Li, and J. Sun, "A comparative study of supervised learning algorithms for re-opened bug prediction," in *Proc. of CSMR*, Genova, Italy, March 2013.
- [19] G. von Krogh, S. Spaeth, and K. R. Lakhani, "Community, joining, and specialization in open source software innovation: a case study," *Research Policy*, vol. 32, no. 7, pp. 1217–1241, 2003.
- [20] B. Shibuya and T. Tamai, "Understanding the process of participating in open source communities," in *Proc. of FLOSS*, Vancouver, Canada, May 2009.

TDR System - A Multi-Level Slow Intelligence System for Personal Health Care

Shi-Kuo Chang, JunHui Chen, Wei Gao and Qui Zhang
Department of Computer Science
University of Pittsburgh, Pittsburgh, PA 15238, USA
{schang, juc52, weg21}@pitt.edu, adora91@gmail.com

Abstract—This paper describes the design of an experimental multi-level slow intelligence system for health care, called the TDR system, consisting of interacting super-components each with different computation cycles specified by an abstract machine model. The TDR system has three major super-components: Tian (Heaven), Di (Earth) and Ren (Human), which are the essential ingredients of a human-centric psycho-physical system following the Chinese philosophy. Each super-component further consists of interacting components supported by an SIS server. This experimental TDR system provides a platform for exploring and integrating different applications in personal health care, emergency management and social networking.

Keywords—slow intelligence system, distributed sensor networks, component-based software engineering.

1. Introduction

Recently there are growing interests in human-centric psycho-physical systems, especially in health care applications. Such human-centric psycho-physical systems have two common characteristics. From the decision-theoretic viewpoint these systems usually have multiple decision cycles such that the actions of slow decision cycle(s) may override the actions of quick decision cycle(s), resulting in poorer performance in the short run but better performance in the long run. From the architectural viewpoint these systems usually have multiple levels to monitor, control and manage many sensors and actuators.

The slow intelligence system is an approach to design such human-centric psycho-physical systems. A slow intelligence system (SIS) is a system that (i) solves problems by trying different solutions, (ii) is context-aware to adapt to different situations and to propagate knowledge, and (iii) may not perform well in the short run but continuously learns to improve its performance over time. The general characteristics of a slow intelligence system include enumeration, propagation, adaptation, elimination, concentration and multiple decision cycles [1]. In our previous work, an experimental test bed was implemented that allows designers to specify interacting components for slow intelligence systems [2].

To facilitate the design of complex slow intelligence systems such as human-centric psycho-physical systems, the concept of super-components is formulated [3]. A complex slow intelligence system basically consists of interacting super-components, which further consists of many interacting components supported by an SIS server. Communications in SIS are through the SIS server and the messages are *layered*, i.e., each message type has its hierarchical *scope*. A super-component can thus be viewed as a collection of components interacting by messages within the same scope. From an architectural viewpoint the result is a multi-level slow intelligence system as illustrated by Figure 1.1.

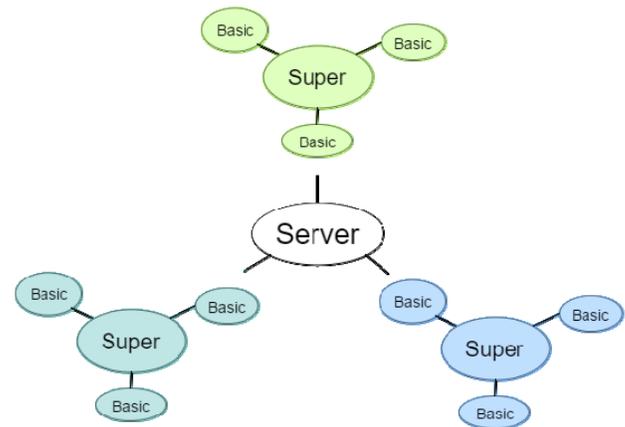


Figure 1.1. A multi-level slow intelligence system.

This paper describes the design of an experimental multi-level slow intelligence system for health care, called the TDR system, which mainly consists of three super components: Tian, Di and Ren. According to the Chinese philosophy these three super-components are the essential ingredients of a human-centric psycho-physical system. They can be thought of as human beings (Ren) interacting with the environment consisting of heaven (Tian) and earth (Di). Decision making in TDR system is through multiple computation cycles involving the super components to increase the chances of survival of human beings. Any action based on only one aspect of the environment without considering the other aspects could reduce the chances of survival, thus iterative, multiple computation cycles are crucial for the TDR system.

The paper is organized as follows. Section 2 presents an abstract machine model for the computation cycles. The TDR system architecture is described in Section 3. The Tian super-component is described in detail in Section 4. Since the Ren super-component has been described in the first author's

previous paper on slow intelligence system for health care [4], it will not be repeated here. A user-friendly GUI for the TDR system is described in Section 5. The TDR system was implemented in Java and GUI implemented in PHP. This test bed for TDR system thus offers an experimental platform for exploring and integrating different applications in personal health care, emergency management and social networking, some of which will be discussed in Section 6.

2. The Abstract Machine Model for Computation Cycles

As mentioned in Section 1 an SIS typically possesses at least two decision cycles. The first one, the *quick decision cycle*, provides an instantaneous response to environmental changes. The second one, the *slow decision cycle*, tries to follow the gradual changes in the environment and analyze the information acquired from the environments or peers or past experiences. The slow/quick decision cycles enable the SIS to both cope with the environment and meet long-term goals.

Complex SISs may possess multiple slow decision cycles and quick decision cycles. Most importantly, actions of slow decision cycle(s) may override actions of quick decision cycle(s), resulting in poorer performance in the short run but better performance in the long run.

To model such decision cycles we introduce an abstract machine model of multiple computation cycles in Section 2.1, and then specify the computation cycles for the TDR system in Section 2.2. In Section 2.3 we describe the steps to compile the abstract machine model into working components of the TDR system.

2.1. The Abstract Machine Model

The Abstract Machine Model is specified by: $(P, S, P_0, \text{Cycle}^1, \dots, \text{Cycle}^n)$, where

- P is the non-empty problem set,
- S is the non-empty solution set, which is a subset of P_0 ,
- P_0 is the initial problem set, which is a subset of P,
- $\text{Cycle}^1, \dots, \text{Cycle}^n$ are the computation cycles.

Each computation cycle will start from an initial problem set and apply different operators such as $+adap_{Aij}$, $-enum$, $>elim$, $=prop_{Aij}$ and $>conc$ successively to generate new problem sets from old problem sets until a non-empty solution set is found. If a non-empty solution set is found, the cycle is completed and later the same computation cycle can be repeated. If on the other hand no solution set is found, a different computation cycle is entered.

As an example the problem set P consists of problem elements $p_1, p_2, p_3, \dots, p_n$, and each problem element p_j is specified by a vector consisting of attributes A_{ij} . A computation cycle x will attempt to find a solution set by first adapting based upon input from the environment: $P^x_0 +adap_{Aij} = P^x_1$ is to adapt

based on attribute A_{ij} , for example, by appending A_{ij} to each element in P^x_0 to form P^x_1 . Then it may try to find related problem elements: $P^x_1 -enum < P^x_2$ where $P^x_2 = \{y: y \text{ is related to some } x \text{ in } P^x_1, \text{ e.g. } d(x,y) < D\}$

Next it may try to eliminate the non-solution elements: $P^x_2 >elim = P^x_3$ where $P^x_3 = \{x: x \text{ is in } P^x_2 \text{ and } x \text{ is in } S\}$

Finally the solution elements (or alert messages if there are nosolutions) may be propagated to peers: $P^x_3 =prop_{Aij} + P^x_4$ is to export/propagate attribute A_{ij} to peers.

Therefore this computation cycle can be specified succinctly as follows: $\text{Cycle}^x [\text{guard } x,y]: P^x_0 +adap_{Aij} = P^x_1 -enum < P^x_2 >elim = P^x_3 =prop_{Aij} + P^x_4$

The above expression is a specification of the computation cycle, not a mathematical equation. This expression should be read and interpreted from left to right.

If P^x_4 is non-empty, the Abstract Machine will complete this cycle of computation and terminate at the end of Cycle^x , and it may later resume at the beginning of Cycle^x . Otherwise P^x_4 is empty and the Abstract Machine will jump to a different Cycle^y . This is specified by $[\text{guard } x,y]$ where x is the current computation cycle if a solution set is found (P^x_4 is non-empty), and y is the computation cycle to enter if no solution set is found (P^x_4 is empty). Before an Abstract Machine completes its current computation cycle, it will propagate the solution set (or alert messages) to its peers.

In the above, the elimination operator can be replaced by the concentration operator, whenever the solution set is not known a priori. The concentration operator applies a predefined threshold to filter out problem elements below the threshold: $P^x_1 >conc = P^x_2$ where $P^x_2 = \{x: x \text{ is in } P^x_1 \text{ and } th(x) \text{ above a predefined threshold } t\}$

2.2. Multiple Computation Cycles of TDR System

For the TDR system, a problem element is a combination of Tian, Di and Ren attributes. Those problem elements that are favorable for human survival are in the solution set S. The problem set P consists of problem elements $p_1, p_2, p_3, \dots, p^n$, and each problem element is specified by a vector consisting of the attributes from Tian (heaven), Di (earth) and Ren (human being), i.e.,

$$p_j = (t1j, t2j, \dots, d1j, d2j, \dots, r1j, r2j, \dots)$$

For example, the Tian attributes t_{ij} are atmospheric variables such as amount of sunlight and water level, the Di attributes d_{ij} are residential variables such as ambient temperature and humidity, and the Ren attributes r_{ij} are personal health indicators such as blood pressure, spo2 value, heart rate, etc.

$$p_j = (\text{sunlight}_j, \text{waterlevel}_j, \text{temp}_j, \text{humidity}_j, \text{bloodpressure}_j, \text{spo2value}_j, \text{heartrate}_j)$$

Initially some attributes may not be assigned any value and some may already have pre-assigned values. After most attributes have been assigned values one can decide whether the problem element is in the solution set. (The simplest case is that each attribute A_{ij} has a solution range R_j , and if every attribute A_{ij} falls within the solution range R_j then the problem element p_j is in the solution set S).

In the TDR system, there are continuous interactions among the three super-components Tian, Di and Ren. Each super-component has its own computation cycle, which is basically the following: Starting from some problem set P_0 , the super-component first adapts to the input from the environment as well as from other peer super-components. It then tries to find related problem elements by enumeration. After those problem elements not in the solution set have been eliminated either using the elimination operator or using the concentration operator, the termination condition can be tested. The termination condition is expressed by [guard x, y] where Cycle x is the current cycle and Cycle y is the cycle to jump to. Whenever one super-component completes its computation cycle, if a solution is found the computation ends, otherwise the control is transferred to the next super-component. Since there are three super-components, we will have three computation cycles.

The Tian super-component has computation Cycle1:
 Cycle1 [guard1,2]: $P^1_0 + adap_{A_{ij}} = P^1_1 - enum < P^1_2 > elim - P^1_3 = prop_{A_{ij}} + P^1_4$

Likewise, the Di super-component has computation Cycle2:
 Cycle2 [guard2,3]: $P^2_0 + adap_{A_{ij}} = P^2_1 - enum < P^2_2 > elim - P^2_3 = prop_{A_{ij}} + P^2_4$

Finally, the Ren super-component has computation Cycle3:
 Cycle3 [guard3,1]: $P^3_0 + adap_{A_{ij}} = P^3_1 - enum < P^3_2 > elim - P^3_3 = prop_{A_{ij}} + P^3_4$

Notice the three computation cycles together form a higher-level computation cycle. High-level computation cycles are essential for a complex human-centric psycho-physical system such as the TDR system. In Section 6 we will discuss applications to personal health care.

2.3. A Compiler for the Abstract Machine Model

The Abstract Machine Model is a formal specification of the computation cycles of a slow intelligence system. Once the abstract machine model is provided, a compiler can be constructed to generate the components. In what follows we describe the major steps of the generic Abstract Machine Compiler (AMC) and the components it generated in pseudo codes.

Step 1: Adapt input from the environment

The AMC will first generate the basic components to gather input from the environment (see box below).

```

Basic Component:
//initialize
threshold = user input();
while (true) {
//adapt input from environment
currentData = collectEnvironmentData();
if (currentData exceed threshold) {
send alert message to Controller and/or Advertiser;
} else {
send normal message to Controller on demand
}
}
  
```

AMC Controller maintains the state and makes decisions based upon different states (see box below).

```

Controller:
//maintain the state within controller. Make decision based on different state.
create and run stateMachine;
while (true) {
msg = getMsgFromSocket();
do something that is not related to state machine
stateMachine.perform(msg); //based on different states, perform
differently when given input
}
  
```

For each Controller, when given some input, the State Machine will determine the action and the output. It may give several tries. For example, two solutions can be applied to one certain state when given certain input (see box below).

```

State Machine:
//define the states
enum Status {
State0,
State1,
...;}
Status currentState = State0; //initial state
void perform(Message msg) {
//based on different states, perform differently when given input
switch (currentState) {
case 'State0':
based on message type and purpose, perform action or
change state
break;
case 'State1':
based on message type and purpose, perform action or
change state
break;
....
}
}
  
```

Step 2: Enumerate and find related problem elements

For Step 2 and Step 3 the AMC is custom designed to handle different patterns from a pattern knowledge-base. For example, if the pattern is “picnic” the initial problem set may be as follows: $P_0 = \{([0,10], [0,10], [10,20], [60,120], [60,80], [50,80])\}$ where $p_j = (flower1_j, flower2_j, temp_j, bp_j, spoj, ekg_j)$.

To answer the question “Is today a good day for picnic?” the temperature sensor is first used to measure temp. Depending on the results of the measurement, either enumeration operator or elimination operator can be applied.

Suppose the temp is 25. Since the temp is normal it cannot be used to eliminate other problem elements and therefore after

enumeration $P1 = \{([0,10], [0,10], 25, [60,120], [60,80], [50,80])\}$. More computation is needed.

Step 3: Eliminate non-solution elements

Suppose the temp is 40. Since the temp is too hot, other problem elements are eliminated and therefore after elimination P1 is empty. Either the conclusion is “today is not a good day for picnic” or another computation cycle may be entered (to find an indoor location for picnic, for example).

Step 4: Propagate solution elements to peers

Once a solution is obtained, the abstract machine will propagate the solution to its peers. For example, several super components may do the work at the same time. Once one super component gets the solution, the rest of them can stop work. An Advertiser will then inform the other super components (see box below).

```

Advertiser pseudo code:
while (true) {
  msg = getMsgFromSocket();
  switch (msg.type) {
    case 'Alert':{
      uploadAlert();//upload alert message to database
      propagateAlert();//propagate alert message to its peers }
    ...
  };
  if( solution_set != null )
    propagateSolution();//propagate solution to its peers
    terminateCycle();//terminate computation cycle}
  else switchCycle();//switch to a new computation cycle }
}

```

3. The TDR System Architecture

As mentioned in Section 1, the TDR system is a multi-level slow intelligence system consisting mainly of three super-components: the Tian super-component, the Di super-component and the Ren super-component. The TDR System architecture is illustrated by Figure 3.1.

The TDR system has a common SIS server to support multi-level messaging. There is an integrated database to store TDR records, and a web GUI that supports the reception and sending of messages. Each super-component has its own sensor(s) to collect information from the environment. For example as shown in Figure 3.1 the Tian super-component has two plant sensors: Parrot Flower 1 and Parrot 2, the Di super-component has an ambient temperature sensor, and the Ren component has a blood pressure sensor. Each super-component furthermore consists of the following components: a monitor component to make sure the information collected by the sensor(s) is within certain acceptable range, a GUI component to interact with the user, an Uploader component to upload the collected information to the next higher level and last but not least a controller component to control the activities of the various components to realize the computation cycles described in the previous section.

When there are multiple controllers in a super-component such as the Tian super-component, a coordinator component can be introduced to coordinate the activities of the controllers and

collect the information provided by the controllers. Generally speaking both the controller component and the coordinator component are essentially controllers, which should possess both the abilities to coordinate and to control the sub-components.

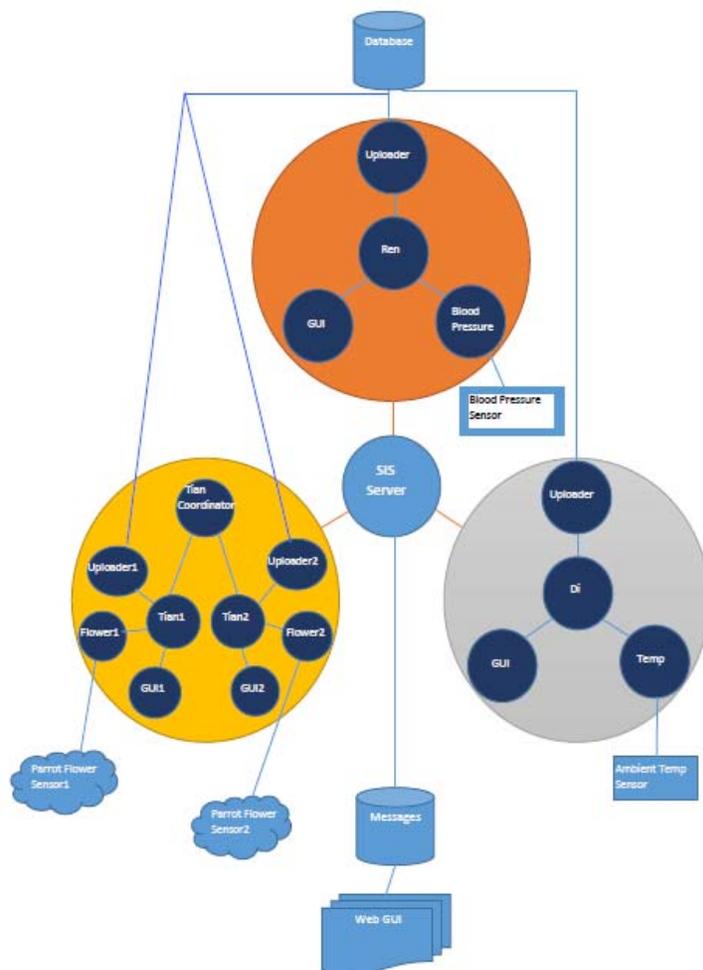


Figure 3.1. The TDR system architecture.

In the experimental test bed, the following functionalities are provided:

3.1 Define a Component

A component should have predefined scope, predefined role and unique name within its predefined scope, these can be described in the component’s Register message, which is stored as an XML document under xml/InitXML. All parameters are defined as Key-Value pairs. The scope defines where outgoing messages from this component can go and the scope of incoming messages this component can receive. Role defines the type of component that can only handle certain types of incoming/outgoing messages. Among all components with the same name within a certain scope, only one of them can be active, i.e., the component name must be unique. There are currently six predefined roles: Basic, Monitor, Advertiser, Controller, Coordinator and Debugger.

3.2 Create a Component

Once the designer knows how a component should behave, he can start to implement it under the Components folder. Components with all kinds of roles have similar templates for implementation. There are two places containing information that the designer should pay attention to: (i) xml/initXML where all predefined Register messages for all available components are stored. (ii) For new components the designer should use the same scope and name in both XML definition and for constants in codes under Components / NEW_COMPONENT_NAME_HERE folder (SCOPE, NAME). The implementation of each role is far from being different from each other. As long as the designer doesn't add extra message types to the collection of acceptable incoming messages, he can simply replace all scopes and names (folder name, SCOPE, NAME, class name, java source code CreateXXX, XML under initXML folder) and create a new working component almost immediately. The six different roles of components are as follows:

3.2.1 Basic Component

For a Basic component such as Blood Pressure in Figure 3.1, no changes are necessary for main method. Method "initRecord" is provided as a place for putting initialization code. Method "componentTask" is provided as a place for putting periodically executed code, such as collecting data. Method "ProcessMsg" is provided as a place for handling different types of messages. Other variables can be added if needed, but the framework should suit the general purpose for implementing a Basic component that sends out Readings which are collected from a data source.

3.2.2 Monitor Component

For a Monitor component such as any GUI in Figure 3.1, it can be designed as a general monitor or a visual console to display data. Other variables can be added if needed.

3.2.3 Advertiser Component

For an Advertiser component such as any Uploader in Figure 3.1, it can be designed as a tool to process the Readings and send anything outside the system via emails, sockets, etc.

3.2.4 Controller Component

For a Controller component such as Ren in Figure 3.1 to process combination of TempBloodPressure measurements, it can be broken down into code segments similar to the TempBloodPressure segment. Five types of code segments are under the ControllerComponents/TempBloodPressure folder: "initial.java" contains all initialization code of extra variables, "helper.java" contains all helper methods used and "helperClass.java" contains all user defined classes. By default Controller components only process Alert messages from Basic components. Alert messages must have unique names.

The same names are used to create code snippets under the TempBloodPressure folder.

3.2.5 Coordinator Component

The Coordinator component such as Tian Coordinator in Figure 3.1 processes the messages from controller components and other components and coordinates the activities of controller components and other components.

3.2.6 Debugger Component

The default Debugger is the PrjRemote.exe tool. It can be replaced by a customized Debugger. However, when a component is assigned the Debugger role, it will get a copy of all messages within the scope that it is in.

3.3 How to Run a Component

Scripts for the Controller component will be automatically generated. For all other roles customized scripts must be provided under the Scripts folder. For Basic or Monitor or Advertiser component, one can simply copy the BloodPressure or the GUI or the Uploader component, respectively, and do some name replacement.

3.4 Scoping

There can be multilevel scopes, each of which contains components that collaborate with each other or are related to each other. Scoping provides a way to further divide the components. By default messages will only be sent within current scope, but one can add ("Broadcast", "True") and ("Direction", "[Up/Down]") to enable broadcast of messages.

3.5 Trouble-Shooting

If a component cannot be connected to the SSISServer, one should check SCOPE and NAME in both code and xml definition. If a message is not delivered, check if the message is sent to a target that does not process this type of message. It is also possible that one forgets to add certain parameters to the message such as valid Scope, Sender, Purpose, etc.

4. The Tian Super-component

4.1 System Structure

A plant is heavily dependent on the environment. According to Chinese philosophy, we may consider the plants' status as Tian (heaven), which will indicate environmental status to some degree. The plant sensors made by Parrot are used in our experiment, which can gather such data as amount of sunshine, moisture, temperature and amount of fertilizer in a plant's environment (see Figure 4.1).

In Tian super-component, we include two different parrot-flower-sensors for plants in different locations. Since they are located in different places, they can gather data from two

different environments. Figure 4.2 illustrates the interactive Tian super-component. Notice there are two Tian controllers coordinated by the Tian coordinator.



Fig 4.1. The plant sensor Parrot Flower Power.

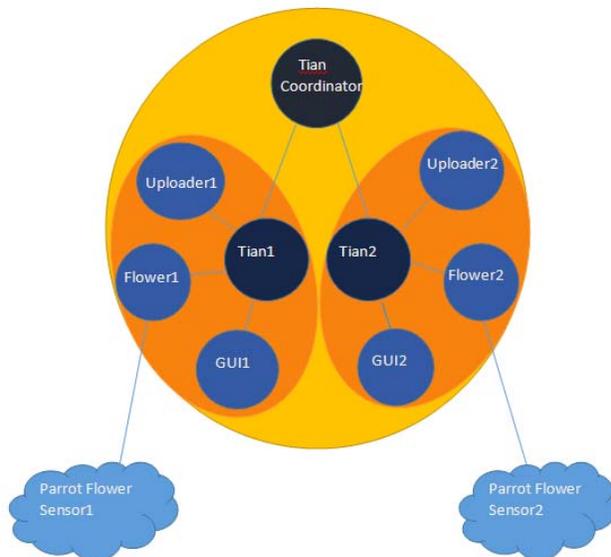


Figure 4.2. The Tian Super-component.

Thus the Tian Super-component has three layers:

- **Top Layer:** Tian coordinator
- **Middle Layer:** Tian1/Tian2 controllers
- **Bottom Layer:** Flower1/Flower2, GUI1/GUI2, Uploader1/Uploader2

Bottom layer is in charge of getting data from the sensor, displaying data to user, and uploading the data to database. Middle layer is in charge of logically activating/deactivating bottom layer components based on instructions from higher level. Top layer is in charge of aggregating data from lower layers. Top layer coordinator will also communicate with other super-components' top-layer.

4.2 Data Path

There are two paths for the data. One is for data going upwards, which is done through Uploader1 and Uploader2. The other is for data going downwards, which is handled by Tian1 and Tian2. Tian1 and Tian2 are both controllers and can make their own decisions such as activate or deactivate the corresponding Flower1 and Flower2 components.

4.3 Control Message Definition

1. Activate all components
 Sender: Web GUI
 Receiver: Tian1/Tian2
 Purpose: activate all components under Tian1/Tian2 sub-system
2. Deactivate all components
 Sender: Web GUI
 Receiver: Tian1/Tian2
 Purpose: deactivate all components under Tian1/Tian2 sub-system
3. Active Flower1/Flower2 component
 Sender: Tian1/Tian2
 Receiver: Flower1/Flower2 component
 Purpose: activate Flower1/Flower2 component
4. Deactivate Flower1/Flower2 component
 Sender: Tian1/Tian2
 Receiver: Flower1/Flower2 component
 Purpose: deactivate Flower1/Flower2 component

4.4 AMC Compiler Steps for Tian

Compared to the generic Abstract Machine Compiler AMC described into Section 2.3, the Tian Compiler has these following steps: Step 1 and Step 4. The other two steps do not exist. In what follows we describe how the Compiler generates the various components in pseudo codes.

In Tian compiler, there are two alert states and four inputs: Alert Flower1, Alert Flower2, Activate all components and Deactivate all components. Since the problem vector will have one and only one state element to be 1 and one and only one input element to be 1, so there are a total of $C(1, 2) * C(1, 4) = 8$ different problem vectors.

For example,
 $p_0 = (1, 0, 0, 1, 0, 0)$.

This specifies when in normal state, given input Alert Flower1, how the abstract machine should perform.

Step 1: Create flower component to adapt input from the environment

The Flower1 and Flower2 monitors will gather environmental data (Sunlight, Moisturizer, Temperature, and Fertilizer) stored in parrot cloud which is updated by parrot sensors. The Flower1 and Flower2 monitors will generate alert message

when new environmental data come in. Only Flower1 component and Tian1 component will be described below.

```
Flower1 Component:
//initialize
while (true) {
//adapt input from environment
currentData1 = collectDataFromSensor1();

//send alert message if exceed threshold
if (currentData1!=null) {
sendAlertMsgTo(Controller);//inform
Controller that new data comes in
sendAlertMsgTo(Advertiser);//inform
Advertiser to propagate new data
} else {
//no new data comes in
}
}
```

```
Tian1:
//maintain the status within controller. Make decision based on different status.
//create and run stateMachine;
msg = getMsgFromSocket();
switch (msg.type) {
case 'Alert':
if(msg.sender == "Flower1"){
Tian1dataArray.add(msg);
sendEmergencyMessageTo("Tian Coordinator");
}
break;
case 'Setting':
stateMachine.perform(msg);
break;
}
}
```

```
StateMachine:
Status currentState = ALERT;//initial state
void perform(Message msg) { //There are two kinds of messages:
//a). environment data message from Tian Components, including alert flower
//b). control message from WebGUI, including activate and deactivate all
components
switch (currentState) {
case ALERT:
switch (msg.type) {
case 'Setting'://control message from webGUI
switch (purpose) {
case 'Activate':
activate Tian components.s
case 'Deactivate':
deactivate Tian components } break; } break;
}
}
}
```

```
Coordinator:
switch (msg.type) {
case 'Emergency':
if(msg.sender == "Tian1"){Tian1Array.add(msg);
}
if(msg.sender == "Tian2"){Tian2Array.add(msg);
}
break;
}
}
```

- Step 2: (does not exist)**
- Step 3: (does not exist)**
- Step 4: Create upload component to propagate solution elements to peers.**

The upload and propagate components can upload necessary messages to the database and propagate to its peers. For example when new environmental data is received an alert message will be sent to database (see box below).

```
Advertiser pseudo code:
while (true) {
msg = getMsgFromSocket();
switch (msg.type) {
case 'NewFlower':{
uploadAlert();//upload alert message to database
propagateAlert();//propagate alert message to its peers
}
...
};
if( solution_set != null ) {color-code("blue");
//this component is color-coded "blue"
propagateSolution();//propagate solution to its peers
terminateCycle();//terminate computation cycle}
else {color-code("red");//this component is color-coded "red"
switchCycle();//switch to a new computation cycle }
}
}
```

5. The Web GUI

The dashboard is the main GUI interface of the TDR system. As illustrated by Figure 5.1 it provides a high-level overview of the data in the system.

On the left side, it has a menu panel that contains all the actions the user can perform, including activating and deactivating components. For the super user, this menu will also include addition, deletion and modification of regular users. The activation and deactivation messages are sent utilizing a message database (MDB) and the TDR components will actively fetch the incoming messages from the MDB (see Figure 3.1).



Figure 5.1. The dashboard for super user.

There is a carousel that displays all components in rotation, four at a time for the PC screen and only one for the smart phone screen. This vividly demonstrates the idea of computation cycles in the TDR system. A component's banner is in *tranquil* state (blue color) until an alert is received and then it changes to red color. Below the carousel panel, a table will be displaying all records that belong to the current user. For each entry, it contains the date and time of a record, the sensor type, the data type, the actual reading of the data, and

the originator. This scheme allows flexibility and scalability, as in the future there might be more and more sensors added to the TDR system.

By clicking the “find similar” button at the lower right part of the dashboard, messages such as M3 will be sent to the MDB to be fetched by the similarity retrieval component to find other user’s records similar to the current user’s record. The messages sent are exactly the same as the ones used in the TDR system, hence the web interface can be viewed as one remote component of the TDR system.

After clicking one specific component on the carousel panel, it will show a detailed list of records that are from the component. If the user is communicating remotely with his/her doctor, a user might want to specify the record ID so that the doctor knows exactly what entry he/she is referring to. A graph showing the data-to-day changes of a selected data item can also be displayed by the GUI for visualization purpose.

6. Discussion

In the formulation of the computation cycles for the TDR system, one can start with the computation cycle of any one of the three super-components. For example the TDR system may start with Ren, i.e., the human conditions are first taken into consideration. Then the atmospheric environmental attributes from Tian and surrounding residential attributes from Di are considered so that an overall solution can be found to enhance the chances of survival of the human being. The TDR system then tries to find appropriate Tian attributes t_{ij} representing atmospheric environmental variables such as sunlight and water level, etc. and Di attributes d_{ij} representing surrounding residential variables such as ambient temperature and humidity, etc.

Alternately the TDR system may also start with the Tian or the Di computation cycles. Different constrained optimization algorithms can be formulated depending on the structure of multi-level computation cycles for Tian, Di and Ren to obtain the “best” solution, i.e., the solution that increase the probability of human survival the most. Finally instead of (or in addition to) constrained optimization algorithms we can also manually set certain variables by human-centric interactions or through social interactions.

One of our main goals is to expand the TDR system for the computation of Chi (also spelled as Qi in Chinese transliteration system HanYu PinYin). The Chi super-component is regarded as at a higher level. It has attributes including both objective measurements and subjective evaluations. Some researchers propose to employ electrical measurements to estimate Chi [5]. Other researchers propose to combine objective measurements with subjective evaluation into an evaluation matrix to estimate Chi [6]. This makes the

Chi super-component both pro-active and adaptive at multiple levels.

The dashboard for TDR system can be further refined. When one clicks on “view details” for the Chi super-component, a list of attributes for Chi is shown. The objective measurements in this list is filled by the multi-level computation cycles based upon actual measurements. The subjective evaluations are entered by the principal user himself/herself based upon his/her subjective feelings.

The experimental TDR system provides a versatile platform for exploring and integrating applications such as personal health care, emergency management and social networking, etc. These applications are currently being investigated at our research laboratory, with major emphasis on an experimental TDR system to estimate Chi. We will also further investigate the theoretical issue to define and characterize the *resonance state* of a system with multiple, multi-level computation cycles.

References

- [1] Shi-Kuo Chang, "A General Framework for Slow Intelligence Systems", *International Journal of Software Engineering and Knowledge Engineering*, Volume 20, Number 1, February 2010, 1-16.
- [2] Shi-Kuo Chang, Yingze Wang and Yao Sun, "Visual Specification of Component-based Slow Intelligence Systems", Proceedings of 2011 International Conference on Software Engineering and Knowledge Engineering, Miami, USA, July 7-9, 2011, 1-8.
- [3] Shi-Kuo Chang, Yao Sun, Yingze Wang, Chia-Chun Shih and Ting-Chun Peng, "Design of Component-based Slow Intelligence Systems and Application to Social Influence Analysis", Proceedings of 2011 International Conference on Software Engineering and Knowledge Engineering, Miami, USA, July 7-9, 2011, 9-16.
- [4] Shi-Kuo Chang, Sen-Hua Chang, Jun-Hui Chen, Xiao-Yu Ge, Nikolaos Katsipoulakis, Daniel Petrov and Anatoli Shein, "A Slow Intelligence System Test Bed Enhanced with Super-Components", Proceedings of 2015 International Conference on Software Engineering and Knowledge Engineering, Pittsburgh, USA, July 6-8, 2015, 51-63.
- [5] Ming-Feng Chen, Hsi-Ming Yu, Shu-Fang Li and Ta-Jung You, "A Complementary Method for Detecting Qi Vacuity", *BMC complementary and alternative medicine*, Vol. 9, No. 12, 2009.
- [6] Ke-Feng Huang, *Effects of Energy Absorption on Meridian System* (能量攝取對經絡系統影響之效應), Doctoral Dissertation (in Chinese), Institute of Biomedical Engineering, National Yang-Ming University, Taiwan, June 2011.

Building a Domain Knowledge Base from Wikipedia: a Semi-supervised Approach

Kai Chen, Xiang Dong, Jiangan Zhu, Beijun Shen
School of Electronic Information and Electrical Engineering
Shanghai Jiao Tong University, Shanghai, China

Email: voyageckg@sjtu.edu.cn, dongxiang@sjtu.edu.cn, jszjgts@sjtu.edu.cn, bjshen@sjtu.edu.cn

Abstract—Knowledge bases are becoming indispensable to software engineering and knowledge engineering. However, the existing domain knowledge bases are always artificially constructed and small-scale. In this paper, we propose a semi-supervised approach to domain concepts detection and software engineering knowledge base construction from Wikipedia. First, the approach selects domain relevant tags from Stackoverflow. Then, it matches Wikipedia entities and expands the concept set through an improved label propagation algorithm. A rule-based method is designed to discover semantic relations including `relate`, `subclassOf` and `equal` by analyzing structural information of Wikipedia. A relation derivation mechanism is presented to optimize the relation set. We finally construct SEBase, a domain-specific knowledge base of software engineering. Experimental results show the high accuracy of the integrated concepts and relations. Compared with other knowledge bases, SEBase has the widest coverage of concepts and relations in software engineering.

Index Terms—Knowledge Base; Software Engineering; Semi-supervised; Domain Concept; Semantic Relation

I. INTRODUCTION

Knowledge bases play an important role in software engineering and knowledge engineering. For example, in program comprehension, knowledge bases are used to compute the semantic similarities between words from the comments and identifiers in software [1]. In software maintenance, knowledge bases provide an effective way to measure the relatedness between documents [2].

Research on general knowledge bases has been quite mature. Some famous achievements like DBpedia [3], Yago [4], WikiTaxonomy [5], BabelNet [6] and Probase [7] are proposed in recent years. A large number of concepts and relations with high accuracy are detected in those knowledge bases. However, the knowledge in them is not specific and not in-depth enough when we focus on some particular domains like software engineering.

Building a domain-specific knowledge base is a difficult task requiring skills in logic and ontological analysis. Domain experts need to determine the scope of the domain concepts and construct relations for them. However, manual approaches are time-consuming and small-scale. Although there has been a considerable amount of prior research on automatic construction of software engineering knowledge base, a high-quality

knowledge base is still lacking because learning knowledge from any single data source cannot achieve desirable effect and domain concepts are difficult to distinguish.

Knowledge learning can be encyclopedic-based or web-based. For the encyclopedic-based approaches, researchers mainly focus on Wikipedia for its abundant structural information. Wikipedia is the most comprehensive and authoritative knowledge source in the world, which has a total of 17 million entries composed of *title*, *redirect title*, *comment*, *text* and other information.

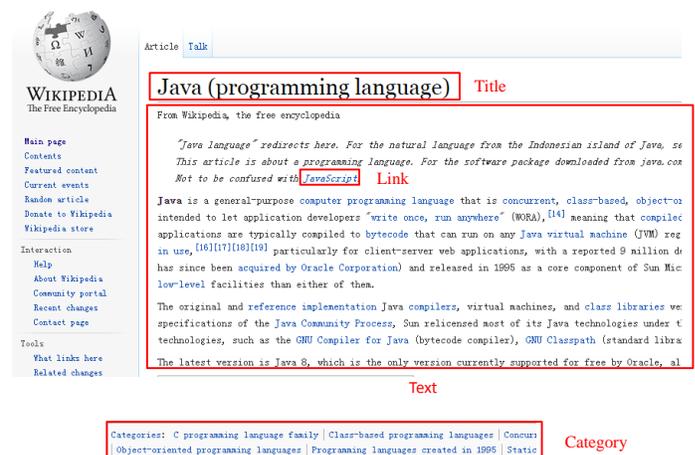


Fig. 1: An example from Wikipedia

For web-based approaches, general knowledge bases learn knowledge from kinds of web pages but domain knowledge bases only consider a part of relevant web pages. In recent years, researchers pay more attention to Stackoverflow. Stackoverflow is one of the most famous QA websites about software engineering and provides a tagging system for users to annotate questions freely. The tags of Stackoverflow are more relevant to software engineering.

Generally speaking, encyclopedic-based approaches can achieve higher accuracy but worse coverage than that of web-based approaches. We select Wikipedia as our main knowledge source and select Stackoverflow as supporting source to obtain the advantages of both. Joorabchi et al. [8] provided a mapping method based on machine learning to interlink Stackoverflow tags with Wikipedia concepts. Unlike their approach, we detect the Wikipedia entities which have high relevance to Stackoverflow tags by matching and expansion methods.



Fig. 2: An example from Stackoverflow

The problem is non-trivial and poses unique technical challenges as follow:

- 1) Since most of the Stackoverflow tags are domain relevant, they are provided by ordinary users freely. A large number of low quality tags exist. If domain relevance of tags cannot be ensured, the accuracy of final results cannot meet our expectations definitely. Therefore, how to ensure the quality of tags remains to be solved.
- 2) Traditional matching methods are difficult to interlink Stackoverflow tags and Wikipedia entities. For example, tag “java” and entity “java man” seem to be related but it is unreasonable to treat the latter as a software engineering concept.
- 3) Although a part of domain concepts can be detected by matching method, more concepts cannot be discovered because tags are not comprehensive enough. For example, “Ocaml” is a programming language but no tag can match it. So how to detect domain concepts which have low relevance to tags is also a challenging problem.

In order to solve challenges above, we decompose the problem into two sub-problems - *domain concepts detection* and *semantic relations discovery*. For the first sub-problem, we select domain relevant tags from Stackoverflow, and make head matching with Wikipedia entities to obtain a part of domain concepts. Then, we treat Wikipedia as a large knowledge network, and propose a semi-supervised method based on an improved label propagation algorithm to acquire domain network of software engineering. For the second sub-problem, we discover three types of semantic relations including relate, subclassOf and equal by analyzing structural information of Wikipedia. We also propose a relation deriving method to delete mistaken and redundant relations.

To the best of our knowledge, our work is the first to build a software engineering knowledge base by interlinking Stackoverflow tags and Wikipedia entities. Our contributions mainly include:

- We systematically explore the knowledge provided by Wikipedia and Stackoverflow, and leverage a series of methods including tag selection, head matching and label propagation to detect domain concepts.
- We analyze structural information of Wikipedia, and dis-

cover three types of semantic relations from Wikipedia. A derivation mechanism is designed to optimize the relation set.

- We carry out a comprehensive set of experiments to evaluate our approach. The results show our approach can outperform the several existing knowledge bases in terms of accuracy and coverage significantly. Thus, we publish SEBase on <https://datahub.io/dataset/se-base> which will benefit many applications in software engineering.

II. RELATED WORK

A. Knowledge Base Construction

Knowledge base construction has aroused extensive attentions in research community. The construction methods can be encyclopedic-based or Web-based. For the encyclopedic-based methods, researchers mainly focus on extracting concept hierarchies from Wikipedia. DBpedia [3] is a crowd-sourced community effort to extract structured information from Wikipedia. WikiTaxonomy [5] builds a taxonomy from the Wikipedia category system. Yago [4] and BabelNet [6] both interlink Wikipedia entities to WordNet [9] synsets.

Regarding Web-based methods, it can be free text based or social tag based. For the free text based methods, Probase [7] builds the largest taxonomy which contains over 2.7 million classes from 1.7 billion Web pages. For social tag based methods, Xiance Si et al. [10] estimated the conditional probability between tags and designed a greedy algorithm to eliminate the redundant relations. Zhishi.schema [11] is the first achievement to learn knowledge from tags and categories in popular Chinese websites.

B. Software Engineering Knowledge Base

Software engineering knowledge base is a type of domain knowledge base. However, there are only a limited number of researches which are related to software engineering knowledge base. Kavi Mahesh et al. [12] proposed LOaD-IT, a concept network to help software developers read technical documents faster. Mr.Izzeddin A.o. el at. [13] constructed a programming language ontology. Software.zhishi.schema [14] uses tags of Stackoverflow build a software engineering knowledge base. We broaden the scale of concepts based on Software.zhishi.schema by interlinking Stackoverflow tags to Wikipedia entities.

C. Set Expansion

The most challenging part to build software engineering knowledge base is domain concepts detection. Set expansion is the most effective method to discover concepts. It can be text-based and graph-based. For the text-based approaches, Richard C. Wang et al. [15] proposed a language-independent set expansion method based on pattern selection. KnowItAll [16] is famous for its high effect. It extracts information from the web and induce the rule templates. Regarding graph-based approaches, label propagation algorithm(LPA) is widely used and it is a semi-supervised method. Jierui Xie et al. [17] [18] focused on community detection using a neighborhood strength

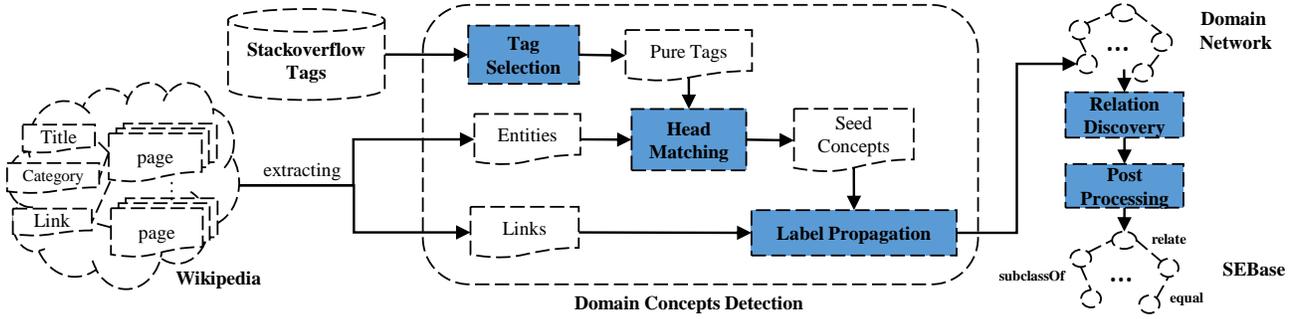


Fig. 3: Our approach to SEBase construction

driven LPA. Liu et al. [19] designed an improved LPA in large-scale bipartite networks to detect community. In this paper, we propose a Wikipedia-based LPA called WLPA to detect community of software engineering.

III. APPROACH

In this section, we describe our proposed approach in detail.

A. Approach Overview

We propose a semi-supervised approach to software engineering knowledge base construction from Wikipedia. As shown in Figure 3, it consists of five main components, namely *Tag Selection*, *Head Matching*, *Label Propagation*, *Relation Discovery* and *Post Processing*. *Tag Selection* tries to select software engineering tags from Stackoverflow. *Head Matching* matches tags with Wikipedia entities to obtain a part of domain concepts. *Label Propagation* leads to discover a domain network including concepts and relations. *Relation Discovery* is to extract three types of relations from the domain network. *Post Processing* is to delete mistaken and redundant relations by three type of relation deductions. Finally we build a domain knowledge base originated from Stackoverflow tags and composed of Wikipedia concepts and relations.

B. Tag Selection

Since tags in Stackoverflow are provided by normal users freely, the domain relevance of tags is unreliable. For example, “music” and “jave” are in the tag set, but the former is domain irrelevant and the latter is a wrong writing of “java”. A selection is necessary to delete domain irrelevant and mistaken tags. The standards of selection are based on two phenomenons. The first is that the tags from questions with high *voting score* and *favorite number* are more domain relevant. This phenomenon can be interpreted as that the users always pay more attention to software engineering related questions. Another truth is that the tags with high frequency are more reliable than tags with low frequency. We select tags that have high occurrence frequency and extract them from questions with high concern rate. The selection method is simple but effective. The details of the selection are as follows:

- 1) We select questions which have the top 10% *voting score* and the top 10% *favorite number*.
- 2) We select tags from those questions and make a ranking by occurrence frequency .

- 3) We select the top 30% tags.

C. Head Matching

The purpose of head matching is to detect Wikipedia entity which has high relevance with the selected tags. The details of the matching are as follows:

- 1) We first unify the format of tags and entities. All letters are transformed into lowercase. A part of tags and entities are split by “-” and the others are split by spaces. We replace underscores and hyphens by spaces. Many tags and entities contain a version number, and we delete it directly because it does not affect semantic meaning. For our running example, “machine-learning” is transformed to “machine learning”, “Visual Studio 2012” is transformed into “visual studio”.
- 2) For each tag and entity, we select its headword as matching standard. We use a simple but effective rule to extract headword: If the tag or entity is a single word, headword is itself. Otherwise, if it contains prepositions such as “of”, “in” or “for”, then headword is the previous word of the preposition. If not then headword is the last word. For example, we extract “sort” from “heap sort” and “architecture” from “logic architecture of hadoop”. It is worth mentioning that some Wikipedia entities such as “java (programming language)” have label. We extract headword from its label. In addition, we use StanfordNLP tool to stem the headword. For some running examples, “programming” is transformed into “program”, “algorithms” is transformed into “algorithm”.
- 3) For each Wikipedia entity, we check if there are one or more tags with same headword. Once the requirement is met, we confirm the entity is a software engineering concept. In order to compare faster, we establish inverse index on headword. We compute digital fingerprint of headword as index.

D. Label Propagation

The head matching only captures the semantic relation between software engineering concepts in an explicit way, however it cannot detect the implicit relations between them. For example, “Ocaml” is a programming language but no tag has semantic relation with it. So “Ocaml” cannot be detected

by head matching, but it is exactly a domain concept. In order to solve this problem, we propose a set expansion method by leveraging structural information to detect the implicit relations between domain concepts.

If we treat *category* and *redirect title* as types of *link*, Wikipedia can be regarded as a large knowledge network composed of entities and links. The linked entity is related to current entity. Correlation degree is proportional to the time of links. We extract links from Wikipedia page and use them to quantitatively characterize the similarities and relatedness between entities. There are more links between concepts in the same domain than that in different domains. In order to expand the domain concepts set, we develop an improved label propagation algorithm (LPA) called WLPA. The details of the algorithm are shown in Table I.

TABLE I: DESCRIPTION OF WLPA ALGORITHM

Algorithm: WLPA
Input: seed concepts set S , Wikipedia entities set W Wikipedia link set L
Procedure:
1: Construct a knowledge network $G(V, E)$. G is directed graph, $V = W$ $edge(u, v) = \infty$ if node u and v connect by redirect title $edge(u, v) = o$ if node u and v connect by category $edge(u, v) = l(u, v)$ if node u and node v connect by link where $l(u, v)$ equals to the number of link from node u to v $\infty \gg o \gg \max\{l(u, v)\}$
2: Initialize nodes with unique labels. $\forall n \in S, c_n = l_Y$ $\forall n \in V \setminus S, c_n = l_N$
3: Update the label of nodes. $\forall n \in V \setminus S, c_n = l_Y$ if $\exists u, c_u = l_Y, e(u, n) = \infty$ or $e(n, u) = \infty$ or $\sum e(u, n) > \sum e(v, n)$ or $\sum e(n, u) > \sum e(n, v)$ where $c_u = l_Y, c_v = l_N, e \neq \infty$ $c_n = l_N$ else.
4: If not converged, continue to 3. (The convergence always be guaranteed because in each iteration, transforming label l_Y into l_N is impossible, due to no more nodes with label l_N generated. The iterations depends on the richness of seed concepts. It typically requires 5 or 6 iterations.)
5: Return final label set $\{c_n\}$
Output: labeled network G

The edges in knowledge network G can be divided into three types that *redirect title* weighs ∞ , *category* weighs o and normal link weighs the number of links. At the initial step, we set nodes of seed concepts with stationary label l_Y . Those nodes would never change its label because they are detected by head matching and we treat them as “seed”. At the propagation step, we derive the label of nodes by comparing the sum of weight between l_Y neighbors and l_N neighbors, where l_Y and l_N indicate that the node is a domain concept or not. The weight of in-edges and out-edges are respectively computed. A propagation running example is shown in Figure 4.

The difference between WLPA and original LPA [20] mainly includes:

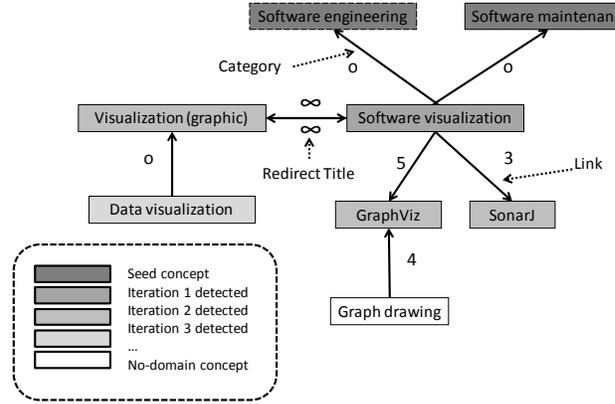


Fig. 4: A running example of label propagation

- Links in Wikipedia knowledge network are directed so we construct a directed graph for WLPA. But original LPA works on undirected graph.
- Links between two entities can be tight or loose. We show this feature by giving edges times of link as weight. But edges of LPA graph do not have any weight.
- In order to ensure the convergence, WLPA sets stationary label for seed concepts.

All of the above changes make WLPA more appropriate for Wikipedia knowledge network. Finally, WLPA generates a domain network composed of software engineering concepts.

E. Relation Discovery

In this subsection, we discover semantic relations from domain network. Three type of relations including *relate*, *subclassOf* and *equal* are extracted. The edges of domain network are composed of Wikipedia structure relations including *link*, *category* and *redirect title*. They are manually constructed by Wikipedia entity contributors. We put those high accuracy relations into our relation sets. The conversion rules are as follows:

TABLE II: STRUCTURE BASED CONVERSION RULES

Situation	Relation
A is B's "Redirect Title"	A equal B, B equal A
A is in B's "Category" set	B subclassOf A

In addition, We use Normalized Google Distance [21] to calculate the degree of correlation between two concepts and construct *relate* relation. Specifically, the Normalized Google Distance (NGD) between two concepts A and B is:

$$NGD(A, B) = \frac{\max\{\log f(A), \log f(B)\} - \log f(A, B)}{\log N - \min\{\log f(A), \log f(B)\}} \quad (1)$$

where N is the total number of edges; $f(A)$ and $f(B)$ are the number of out-edge of concepts A and B, respectively; $f(A, B)$ is the number of nodes that both A and B out-connected.

In order to control the number of this kind of relations, we select five of the most relevant concepts for a concept.

F. Post Processing

We provide a derivation mechanism to delete mistaken or redundant relations. The relation deduction rules are shown in Table III, where $A=B$ means that there is an `equal` relation between A and B, $A\approx B$ means that there is a `relate` relation from A to B, $A\rightarrow B$ means that there is a `subclassOf` relation from A to B.

TABLE III: DERIVATION RULES

Situation	Type	Operation
$A\rightarrow B, B\rightarrow C, A\rightarrow C$	Transitive Redundancy	Delete ($A\rightarrow C$)
$A=B, A\approx B$	Synonym Conflict	Delete ($A\approx B$)
$A=B, A\rightarrow B$	Synonym Conflict	Delete ($A\rightarrow B$)
$A\rightarrow B, A\approx B$	Synonym Conflict	Delete ($A\approx B$)

IV. EVALUATION

In the version of the data we downloaded, Stackoverflow has 38,205 tags while Wikipedia has about 17,000,000 entities. Each entity has an average of 45.2 links, 1.8 categories and 0.23 redirect titles. We finally obtain 128,674 concepts and 607,341 relations about software engineering. In this section, we show the experimental results of our proposed approach. We mainly focus on accuracy and coverage of concepts and relations in SEBase.

A. Selected Tag Quality Evaluation

We first evaluate the quality of selected Stackoverflow tags. The accuracy evaluation has to be done by persons manually because of semantic comprehension required. However, due to the large number of concepts and relations, it is impossible to evaluate all of them by hand. Therefore, we design a random sampling strategy and a labeling process. We manually annotate some targets of sampled concepts or relations. The accuracy assessment on the sampled subset can further be used to approximate the correctness of the whole set. Three students from our laboratory are invited to participate in the labeling process. We provide them with three choices namely *agree*, *disagree* and *unknown* to label each sample. Then we can compute the average accuracy.

We generate four tag sets for comparison. They are original set, question selected set (QS), frequency selected set (FS), question and frequency selected set (QS+FS). We randomly extract 500 tags from four tag set as samples. Students annotate each tag independently. Then, we select 1,000 high correlated tags from original set to compute the coverage of four tag set. Because the tags are treated as seed, we consider the precision is more important than the recall. $F_{0.5}$ score is used to compare comprehensive quality of four sets. The results are evaluated in terms of precision, recall and $F_{0.5}$, as shown in Figure 5 where we can confirm the necessity of *Tag Selection*.

After question selection and frequency selection, the $F_{0.5}$ score achieves 92.82%.

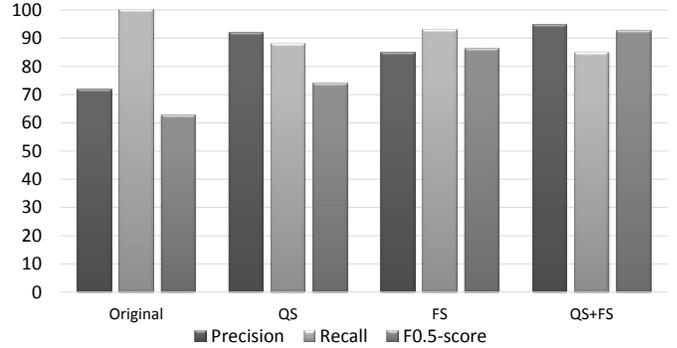


Fig. 5: Tag quality evaluation

B. Seed Concepts Quality Evaluation

This experiment is to evaluate the performance of *Head Matching*. We generate three matching results for comparison. They are original result, head matching result and stemmed result. Original result is generated by matching tags and entities directly. Head matching result is generated by matching headwords. Stemmed result is generated by using stemmed headwords. The sampling is similar to tag quality evaluation. The results are evaluated in terms of precision, recall and F1-score, as illustrated in Figure 6.

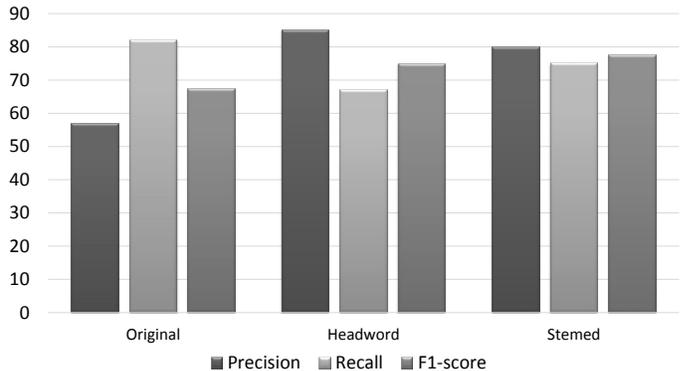


Fig. 6: Seed concepts quality evaluation

The evaluation shows the necessity of *Head Matching*. The precision of directly matching is less than 60% despite the recall is high. Head matching improves precision to 82.4%. The stemming operation balances precision and recall, and therefore the F1-score reaches 77.42%.

C. Final Results Evaluation

Next we evaluate the final concept and relation results. The sampling of concepts is similar to that of the evaluation above. For relations, we randomly extract 500 `equal`, 500 `relate` and 500 `subclassOf` relations as samples. Students label each relation independently. Then, we select same numbers of correct relations from Wikipedia to compute the coverage. The evaluation results are shown in Figure 7.

For concepts, our approach achieves precision of 76.7%, and recall of 92.4%. For relations, our approach achieves expected precision but unsatisfactory recall. This is because we mainly focus on domain concept detection and discover relation from

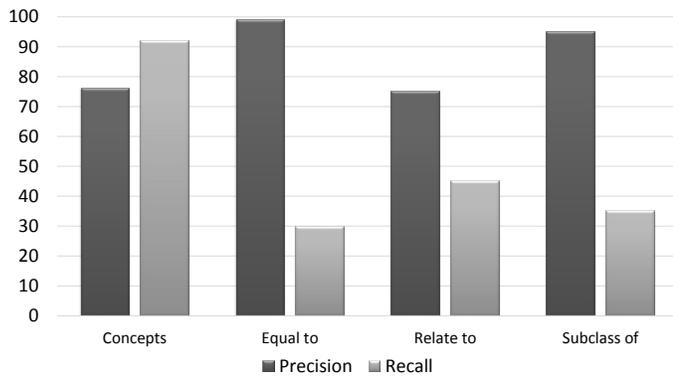


Fig. 7: Final results evaluation

Wikipedia structure. However, only 27% of Wikipedia entities have *category* and 23% of entities have *redirect title*.

D. Comparison with Other Datasets

Since there are no published famous software engineering knowledge bases, we compare SEBase with the subsets about software engineering extracted from other well-known general taxonomies namely Yago and WikiTaxonomy. We also compare SEBase with our previous work, Software.zhishi.schema (SZS), which is constructed from Stackoverflow. The comparison results are shown in Table IV.

TABLE IV: COMPARISON WITH OTHER TAXONOMIES

	SEBase	SZS	Yago	WikiTaxonomy
Concept Number	128,674	38,205	898	711
Concept Overlap	15,441	0	29	27
Subclass of	77,204	68,098	870	630
Equal to	15,441	0	29	27
Relate to	514,696	0	0	0

As for the relation number, SEBase is much larger than any other datasets. However, 76.2% relations of SEBase are *relate* and *subsumption* relations are relatively too few. Regarding to the granularity and richness of concepts, SEBase is more fine-grained than other existing datasets.

V. CONCLUSION AND FUTURE WORK

In this paper, we build SEBase, a large-scale knowledge base of software engineering which contains 128,674 concepts and 607,341 relations. Our approach interlinks Stackoverflow tags and Wikipedia entities, and uses a semi-supervised WLPA to learn knowledge from Wikipedia. The experiments show the high quality of concepts and relations in SEBase.

For future work, we plan to improve SEBase from three aspects. We will try to extract more hyponymy relations based on classifier. On the other hand, we will attempt to deep learning technology to obtain purer tags. Moreover, it is would be interesting to improve matching method more effective by keyword extraction and topic model.

VI. ACKNOWLEDGEMENT

Beijun Shen is the corresponding author. This research is supported by 973 Program in China (Grant No. 2015CB352203) and National Natural Science Foundation of China (Grant No. 61472242).

REFERENCES

- [1] G. Sridhara, E. Hill, L. Pollock, and K. Vijay-Shanker, "Identifying word relations in software: A comparative study of semantic similarity tools," in *Program Comprehension, 2008. ICPC 2008. The 16th IEEE International Conference on*, pp. 123–132, IEEE, 2008.
- [2] R. Wu, H. Zhang, S. Kim, and S.-C. Cheung, "Relink: recovering links between bugs and changes," in *Proceedings of the 19th ACM SIGSOFT symposium and the 13th European conference on Foundations of software engineering*, pp. 15–25, ACM, 2011.
- [3] S. Auer, C. Bizer, G. Kobilarov, J. Lehmann, R. Cyganiak, and Z. Ives, *Dbpedia: A nucleus for a web of open data*. Springer, 2007.
- [4] F. M. Suchanek, G. Kasneci, and G. Weikum, "Yago: a core of semantic knowledge," in *Proceedings of the 16th international conference on World Wide Web*, pp. 697–706, ACM, 2007.
- [5] S. P. Ponzetto and M. Strube, "Wikitaxonomy: A large scale knowledge resource..," in *ECAI*, vol. 178, pp. 751–752, 2008.
- [6] R. Navigli and S. P. Ponzetto, "Babelnet: Building a very large multilingual semantic network," in *Proceedings of the 48th annual meeting of the association for computational linguistics*, pp. 216–225, Association for Computational Linguistics, 2010.
- [7] W. Wu, H. Li, H. Wang, and K. Q. Zhu, "Probase: A probabilistic taxonomy for text understanding," in *Proceedings of the 2012 ACM SIGMOD International Conference on Management of Data*, pp. 481–492, ACM, 2012.
- [8] A. Joorabchi, M. English, and A. E. Mahdi, "Automatic mapping of user tags to wikipedia concepts: The case of a q&a website–stackoverflow," *Journal of Information Science*, p. 0165551515586669, 2015.
- [9] G. A. Miller, "Wordnet: a lexical database for english," *Communications of the ACM*, vol. 38, no. 11, pp. 39–41, 1995.
- [10] X. Si, Z. Liu, and M. Sun, "Explore the structure of social tags by subsumption relations," in *Proceedings of the 23rd International Conference on Computational Linguistics*, pp. 1011–1019, Association for Computational Linguistics, 2010.
- [11] H. Wang, T. Wu, G. Qi, and T. Ruan, "On publishing chinese linked open schema," in *The Semantic Web–ISWC 2014*, pp. 293–308, Springer, 2014.
- [12] K. Mahesh, A. Nagarajan, A. R. Balevalachilu, and K. Prasad, "Curating semantic linked open datasets for software engineering," in *Proceedings of the 2013th International Conference on Posters & Demonstrations Track-Volume 1035*, pp. 61–64, CEUR-WS. org, 2013.
- [13] I. Abuhassan and A. M. AlMashaykhi, "Domain ontology for programming languages," *Journal of Computations & Modelling*, vol. 2, no. 4, pp. 75–91, 2012.
- [14] J. Zhu, H. Wang, and B. Shen, "Software. zhishi. schema: A software programming taxonomy derived from stackoverflow," in *14th IEEE International Semantic Web Conference (ISWC)*, IEEE, 2015.
- [15] R. C. Wang and W. W. Cohen, "Language-independent set expansion of named entities using the web," in *Data Mining, 2007. ICDM 2007. Seventh IEEE International Conference on*, pp. 342–350, IEEE, 2007.
- [16] S. Schiff, "Know it all," *The New Yorker*, vol. 31, no. 07, 2006.
- [17] J. Xie and B. K. Szymanski, "Community detection using a neighborhood strength driven label propagation algorithm," in *Network Science Workshop (NSW), 2011 IEEE*, pp. 188–195, IEEE, 2011.
- [18] J. Xie and B. K. Szymanski, "Towards linear time overlapping community detection in social networks," in *Advances in Knowledge Discovery and Data Mining*, pp. 25–36, Springer, 2012.
- [19] X. Liu and T. Murata, "Community detection in large-scale bipartite networks," in *Web Intelligence and Intelligent Agent Technologies, 2009. WI-IAT'09. IEEE/WIC/ACM International Joint Conferences on*, vol. 1, pp. 50–57, IET, 2009.
- [20] K. Kothapalli, S. V. Pemmaraju, and V. Sardeshmukh, "On the analysis of a label propagation algorithm for community detection," in *Distributed Computing and Networking*, pp. 255–269, Springer, 2013.
- [21] R. L. Cilibrasi and P. Vitanyi, "The google similarity distance," *Knowledge and Data Engineering, IEEE Transactions on*, vol. 19, no. 3, pp. 370–383, 2007.

A Knowledge Modeling Framework for Computational Materials Engineering

Raghavendra Reddy Yeddula, Sushant Vale, Sreedhar Reddy, Chetan P. Malhotra, B. P. Gautham, Pramod Zagade
TRDDC, Tata Consultancy Services Limited
Pune, India

Abstract— Integrated computational materials engineering (ICME) is a new approach to the design and development of materials, manufacturing processes and products. The approach proposes using a combination of modeling and simulation, data-driven reasoning and knowledge-guided decision making. Industrialization of this approach requires strong automation support. Modeling and simulation is a highly knowledge-intensive activity and integrated design requires knowledge cutting across several design domains. For the industrialization vision to succeed, it is essential to capture this knowledge and make it available in a usable form for people not so skilled in these areas. With this motivation, we are building a comprehensive computational platform to support this emerging design paradigm. We present an overview of the platform and discuss its knowledge engineering requirements. We present a knowledge modeling framework to address these requirements.

Keywords— ICME; Knowledge Engineering; Knowledge Representation; Ontology;

I. INTRODUCTION

Integrated computational materials engineering (ICME) is a new method for integrated design of materials, products and manufacturing processes using modelling and simulation, knowledge-guided decision making and data-driven reasoning [1,2]. ICME is widely recognized as a paradigm changer that is expected to significantly reduce the dependence on trial and error based experimentation cycles, leading to a) faster development of new materials, and b) significant improvement in quality and time-to-market of products by integrating material design with product design. However, industrialization of this approach has many roadblocks to overcome [3]. Modelling and simulation is a highly knowledge-intensive activity. In an integrated design, one has worry about a multitude of phenomena occurring at different length scales. Choosing right models for these phenomena, at right scales, with right parameters, and ensuring integration across these models is a non-trivial task. It calls for a great deal of domain knowledge and expertise. Similarly, developing right design workflows also calls for extensive domain knowledge and expertise. Shortage of skilled people in these areas can be a significant roadblock for industrial adoption of ICME. For the industrialization vision to succeed, it is essential to capture this knowledge and make it available in a usable form for people not so skilled in these areas. Hence, a software platform that supports ICME must have a strong knowledge engineering capability. It should capture knowledge about a variety of

material systems, products and processes, etc., in the form of building-block models, design templates, workflows, rules, etc. It should use this knowledge to systematically guide a designer towards right solutions. With this motivation, we are building a comprehensive computational platform called PREMAP [4] that supports concurrent design of materials, manufacturing processes and engineered components.

Fig. 1 shows a view of the architecture of the platform from the knowledge engineering perspective. The architecture supports three user roles, namely, knowledge engineer, design solution builder and designer. The design solution builder is responsible for building design solutions for various engineering problems. A designer uses these solutions to solve his/her design problems. The knowledge engineer is responsible for knowledge management: modeling the knowledge space, capturing and curating knowledge from different sources such as domain experts, literature, etc.

At the heart of the platform we have a knowledge repository that contains engineering design knowledge in the form of models, workflows, design templates, design rules, design cases, etc. A design solution builder consults this knowledge base while building his solutions. The knowledge engineering component plays an active role in this consultation process. It processes the design context to infer knowledge that is suitable for the problem context and offers context-specific guidance. This could be in the form of design guidelines, design templates, workflows, models, past design cases and so on.

For instance, building a solution for designing a gear consists of building a gear component model, developing a design process comprising steps such as geometry design, material selection, manufacturing process design, etc., selecting simulation models, defining services to carry out computations, specifying rules to guide design decisions, and so on. Knowledge-guided assistance plays a key role at each stage in this process. The effectiveness of this guidance depends on the quality of the underlying knowledge engineering framework.

The rest of the paper is organized as follows. In section 2 we identify the requirements of the knowledge engineering framework. In section 3 we present a knowledge modeling framework to address these requirements. In section 4 we discuss how this model supports context-sensitive knowledge retrieval. In section 5 we discuss related work, and in section 6 we summarize our contribution.

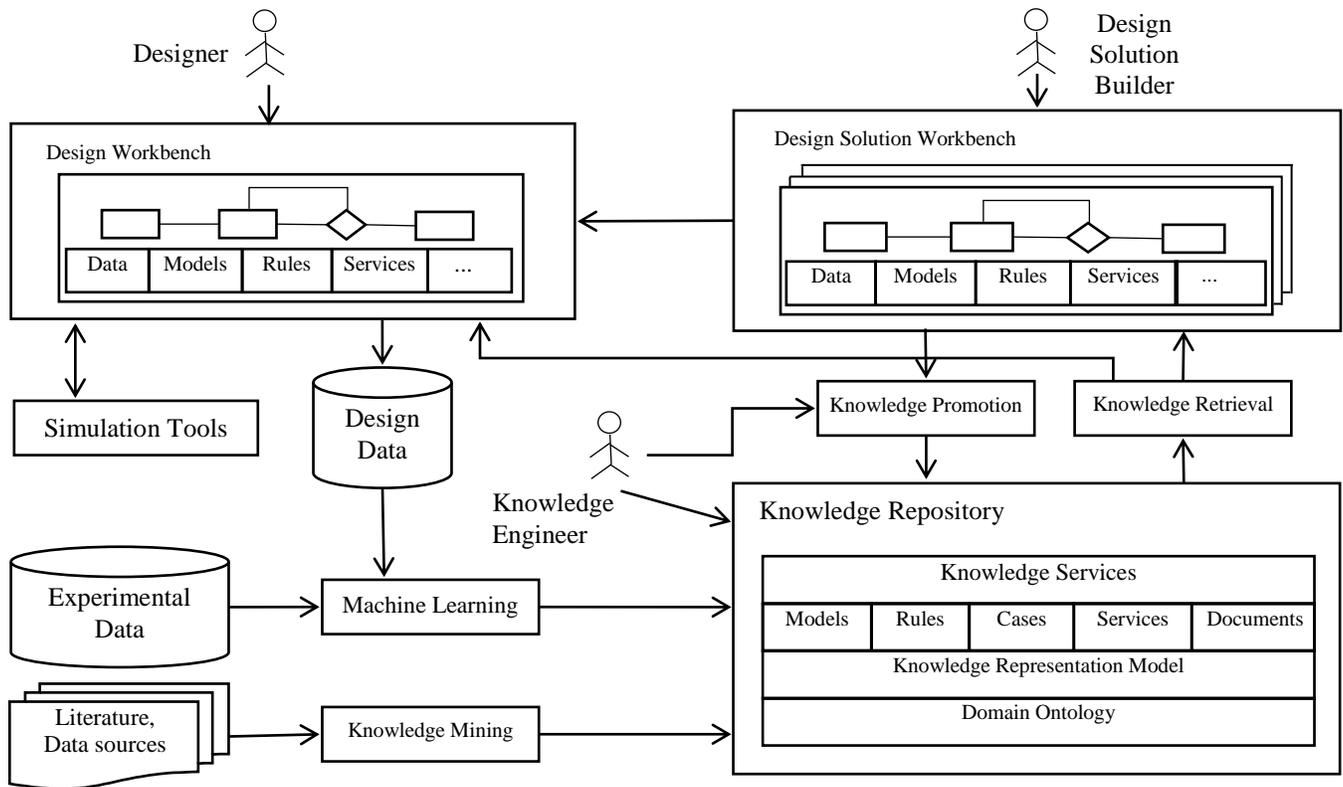


Figure 1. PREMAP architecture from Knowledge Engineering Perspective

II. PREMAP KNOWLEDGE ENGINEERING FRAMEWORK

To effectively serve the knowledge needs of a design platform such as PREMAP, the knowledge engineering framework has to satisfy several requirements:

1. PREMAP is designed to support integrated design cutting across a number of design domains. These design domains are not always known a priori. Existing design domains also change quite frequently. For instance, there may be a new material that needs to be added or a new manufacturing technique that needs to be supported. The knowledge engineering framework should be flexible to support this.

2. In ICME, knowledge exists in multiple forms such as models, rules, cases, design templates, documents and so on. Some of this knowledge is in executable form with automated reasoning (e.g. rules, models, etc.), while some of it is in the form of documents and media that can only be leveraged by a human. It should be possible to represent both kinds of knowledge.

3. Knowledge is intended to serve a certain purpose. It is essential to model this intent space, so as to be able to capture right kind of knowledge for the right purpose.

4. Knowledge is applicable in a certain context. It is essential to capture this context.

5. Knowledge retrieval should be context-sensitive. I.e. retrieved knowledge should be right for a given problem context.

6. Knowledge reuse should be facilitated. I.e. wherever possible, knowledge from other similar problem contexts should be made available for reuse.

7. Knowledge comes from a variety of sources such as domain experts, literature, online sources, and so on. A lot of knowledge also lies hidden in data. It should be possible to mine knowledge from all these different sources.

We use a model-driven approach to realize the knowledge engineering framework. The starting point is a model of the knowledge space that specifies what kinds of knowledge should be captured and how it should be captured. The knowledge space model again has two parts to it: 1) domain ontology that specifies the conceptual space of the domain, and 2) knowledge representation model.

In this paper, we do not discuss the knowledge mining part (requirement 7).

III. KNOWLEDGE MODELING FRAMEWORK

A. Ontology Model

Domain ontology provides the semantic foundation for capturing knowledge. To maximize reuse and utility of knowledge, the conceptual space of a domain has to be modelled at the right level of abstraction. So, getting the domain ontology right is the first step towards modelling the knowledge right. However, ontology varies from subject to subject, and, being a generic platform, PREMAP has to cater to a wide range of subjects. For instance, the ontology of steel is

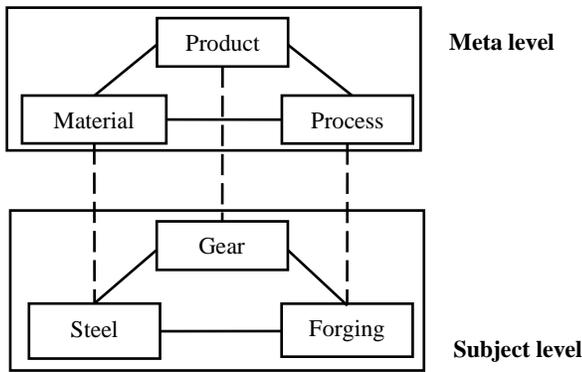


Figure 2. Domain Ontology Layers

different from the ontology of a composite material. This calls for a flexible ontology engineering framework that enables us to create and evolve subject specific ontologies without hard coding them into the platform. To address this, we have conceptualized domain models at two ontological levels - a meta level and a subject level as shown in Fig. 2.

Models relevant for ICME domain can be broadly categorized into three subject areas - materials, products and processes. Corresponding to these subject areas we have three related meta-models -- material meta-model, component meta-model and process meta-model. Subject specific ontologies are created as instances of these meta-models. This layered architecture provides two benefits: 1) Since all subjects of a subject area are described using the same meta-model, the meta-model provides the semantic basis for discovering relations among these subjects, and 2) It lends extensibility to the platform, by enabling new subjects to be created as instances of meta-models. Subject-specific ontologies thus become first class entities in the platform. To achieve this flexibility, we build our ontology modeling framework on a reflexive modeling framework [5] where meta-classes can be used to build multiple modeling levels.

Fig. 3 shows the ontology of Gear as an instance of the component meta-model. A meta-model specifies a set of meta-classes and their relations. A meta-class is a class whose instances are also classes. The figure shows classes in <class>:<meta-class> format and objects in <object>:<class>

format. In the figure, the component meta-model has meta-classes *Component*, *GeometricFeature*, *FunctionalFeature* and *Parameter*. The Gear ontology is created as an instance of this meta-model. A gear is a component whose geometry has features such as hub, web, rim and teeth. Its function is to transmit motion, with a change in rotational speed. Accordingly, the *Gear* class is an instance of the meta-class *Component*, *Hub* class is an instance of the meta-class *GeometricFeature*, and so on. The geometric feature '*Hub*' has *diameter* and *width* as parameters. The figure also shows a specific gear (NanoCarGear) with its dimensions, as an instance of the gear ontology.

Reasoning with the Ontology

Classification is one of the key reasoning capabilities in an ontology. Given a description of a concept, the reasoner should automatically discover concepts that generalize the given concept (subsumption relation). In the context of knowledge engineering, we want to be able to generalize knowledge and reuse it in multiple contexts. The ability to automatically discover generalization relationships is therefore a fundamental capability. We use a description logic (DL) [6] reasoner for this purpose. From subject-specific ontology descriptions, we automatically generate the equivalent DL fragments. This is achieved by specifying generation templates at the meta-class level in the meta-model. For example, to generate DL fragments for components, we attach the following template with the meta-class 'Component' in the meta-model shown in Fig. 3.

```
[template DLOfComponent(c: Component)]
[c.name/] =
  Component
  and
  [for (ff:FunctionalFeature | c.functionalFeature)
separator('and')]
    exists functionalFeature.{{ff.name/}}
  [/for] and
  [for (gf:GeometricFeature | c.geometricFeature)
separator('and')]
    exists geometricFeature.{{gf.name/}}
  [/for]
[/template]
```

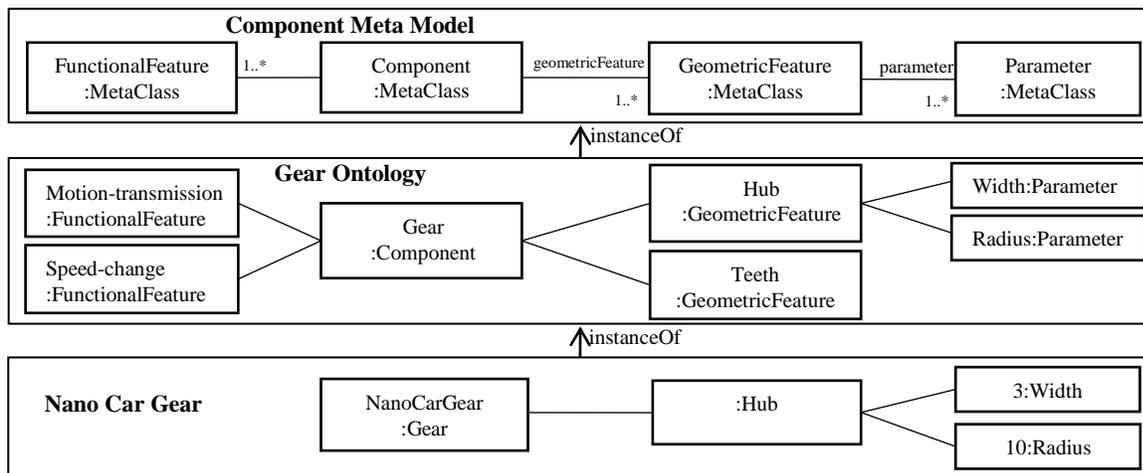


Figure 3. Component Modeling Layers

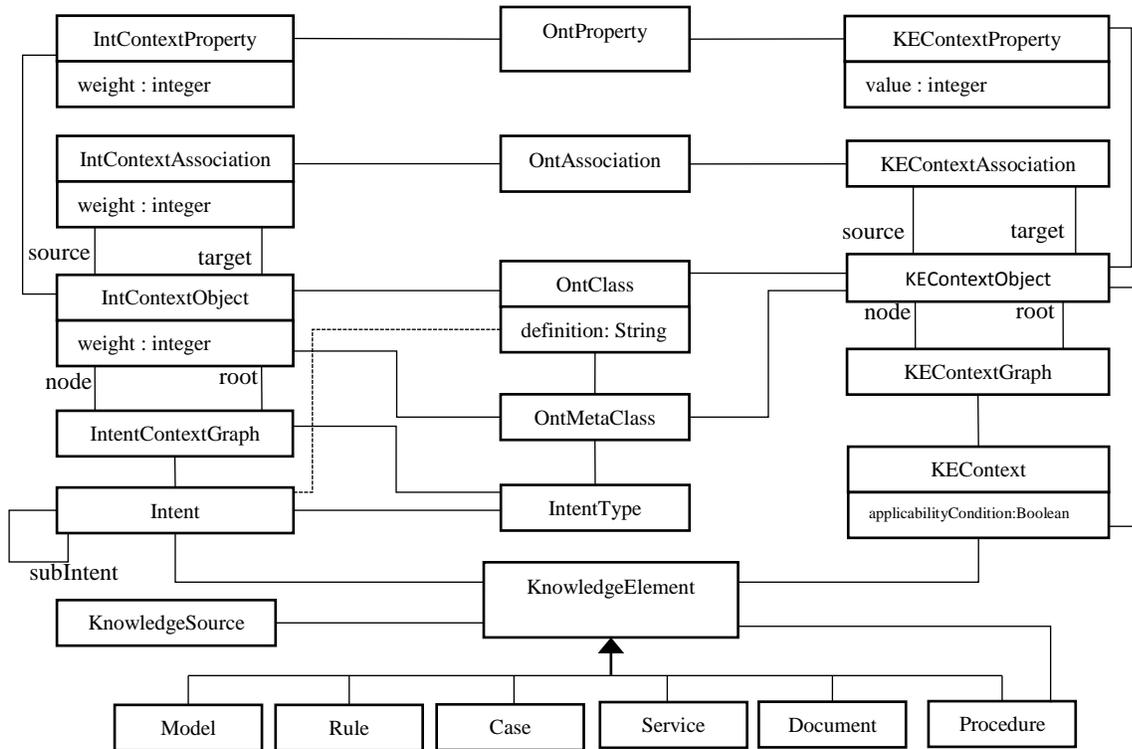


Figure 4. Knowledge Representation Model

The template is specified in Mof2Text specification language [14]. When this template is executed on the gear ontology fragment shown earlier, it generates the DL definition shown below.

```
Gear =
  Component and exists functionalFeature.{Motion-transmission}
  and exists functionalFeature.{Speed-change}
  and exists geometricFeature.{Hub} and exists
  geometricFeature.{Teeth}
```

B. Knowledge Representation Model

Fig. 4 gives an overview of the knowledge representation model. It has four parts: domain ontology, intent model, knowledge element model and knowledge element context model.

The domain ontology model was discussed in the previous section. Ontology provides the domain vocabulary to represent different knowledge elements. Ontology also serves to specify the contexts in which those knowledge elements are applicable. In Fig. 4, *OntMetaClass* refers to an ontology meta-class (e.g. Component), *OntClass* refers to an ontology class (e.g. Gear), *OntAssociation* refers to an ontology association and *OntProperty* refers to an ontology property.

1) Intent Model

Knowledge is required to serve a certain intended purpose in the domain. For example, knowledge to guide requirement specification of a component, knowledge to design a process, knowledge to select a material, and so on. It is essential to model these intents (or topics) in order to ensure that right knowledge is captured for the right purpose. We identify

intents at two levels. At the meta-level, the model element *IntentType* specifies intents of a meta-class. At the subject level, the model element *Intent* specifies intents of a subject-specific class. For example, the meta-class Component has intent types ‘Geometry design’ and ‘Material selection’; the meta-class ManufacturingProcess has the intent type ‘Process design’. At the subject level, the class Forging has ‘Forging process design’ as an intent. This is an instance of the intent type ‘Process design’ of the meta-class ManufacturingProcess. An intent may have sub intents. For instance, forging process design has preform design and die design as sub intents.

An intent may also identify the information context necessary to make decisions about the intent. For example, forging process design decisions depend on the geometry of the component, its performance requirements and the material used. An intent specifies its information context by means of a context graph that identifies a set of connected ontology elements that together provide the information context for the intent. Nodes of the graph identify ontology objects and edges identify the associations between them. Fig. 5 shows the intent context graph for forging process design. An intent's context

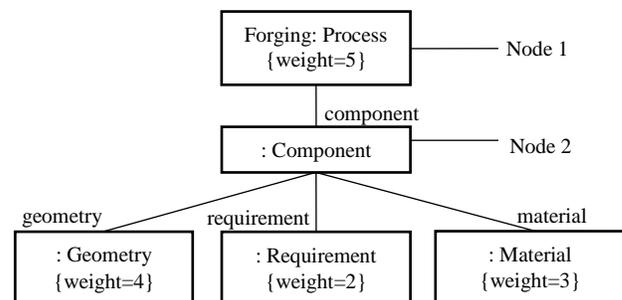


Figure 5. Intent Context Graph for Forging Process Design Intent

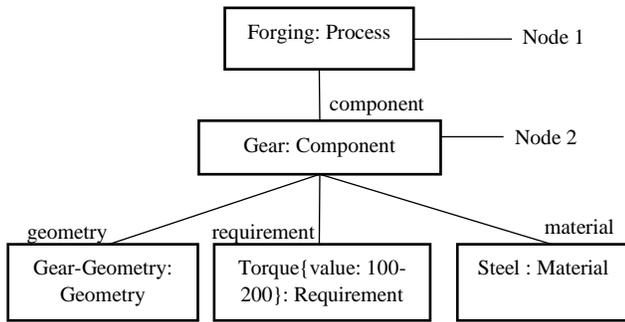


Figure 6. Query graph for fetching forging process design knowledge graph is inherited by all its sub intents, unless explicitly overridden.

An intent context graph provides the template to construct knowledge queries to fetch knowledge for a given intent. The query is constructed from the information available in the problem context. For example, in a problem context where gear is the component, steel is the material and the required torque range is 100-200, the query graph shown in Fig. 5 is constructed from the context graph shown in Fig. 5. An intent context graph may also specify weights for objects, associations and properties, signifying their relative weightages during query processing. The graph in Fig. 5 specifies that a match on geometry has higher weightage than a match on material.

An object node in an intent context graph may specify a class or a meta-class. When a meta-class is specified, the corresponding node in the query graph can be instantiated with any class of the meta-class. This makes a context graph generic and reusable across multiple problem contexts. For example, in Fig. 5, object node 2 specifies the meta-class 'Component'. When constructing the query graph for forging of a gear component, this node would be instantiated with the class 'Gear'; when constructing the query for a clutch component this would be instantiated with 'Clutch', and so on.

2) Knowledge Element Model

A *KnowledgeElement* is a unit of knowledge that serves an intended purpose in a context. Knowledge elements can be represented in multiple forms such as models, rules, cases, services, documents, etc. Of these, models, rules, cases and services are executable knowledge elements, whereas documents are for human consumption. A special knowledge element called 'Procedure' enables composition of multiple knowledge elements into a step-by-step procedure. The model also tracks the provenance of knowledge elements (e.g. from which domain expert, from which publication, etc.).

3) Knowledge Element Context Model

Knowledge is generated in a certain context. It is essential to capture this context to ensure that right knowledge is applied in the right context. We model the context by means of knowledge element context graphs. A knowledge element context graph is essentially a description of the state of the world in which a knowledge element is applicable, expressed as a set of conditions on ontology classes and meta-classes. The graph specifies a set of ontology objects (nodes), their property values and their associations (edges). Its semantics is that the

knowledge element is applicable in the context of the given ontology objects when they are related to each other in the specified manner and when they have the specified property values. A knowledge element context may additionally specify a boolean expression (applicabilityCondition in the model) that these ontology elements must satisfy. In the context graph, an object node may specify a class or a meta-class; when a meta-class is specified, the knowledge element is applicable in the context of 'any class' of the meta-class.

Example 1:

The context graph of knowledge element 'Material Selection Rules for Gear' is shown in Fig. 7. This graph specifies that the knowledge element is applicable for material selection for automobile gears.

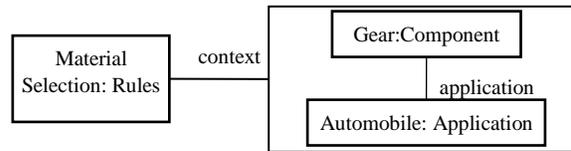


Figure 7. Knowledge element context graph for gear material selection

Example 2:

The context graph for knowledge element 'Billet volume estimation procedure' is shown in Fig. 8. This graph specifies that the billet volume estimation procedure is applicable when forging steel components with axisymmetric geometry.

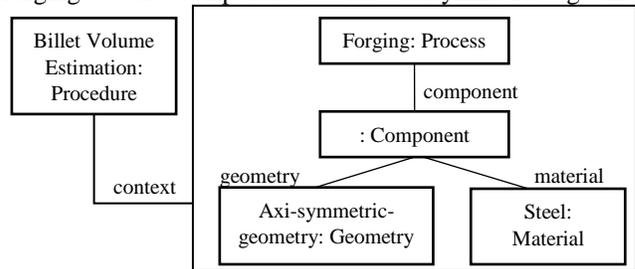


Figure 8. Knowledge element context graph for billet volume estimation procedure

IV. CONTEXT SENSITIVE KNOWLEDGE RETRIEVAL

Knowledge retrieval is done by matching the problem context with contexts stored in the knowledge repository and fetching knowledge elements from matching contexts. This is done by constructing a query graph from the current context as specified by the intent context graph.

To give an example, we consider the gear design case. To start with, we assume that a design solution builder has already developed the gear ontology and a partial design workflow consisting of requirements capture, geometry design, material selection, manufacturing process selection, etc. Let's also assume that the gear is required to handle a torque range of '100 – 200' and the material selected is steel. Let's assume that the solution builder is currently working on designing a forging process for the gear, and in particular on estimating the initial billet volume which is one of the sub steps of the design process.

So the current intent is 'Estimate Initial Billet Volume' which is one of the sub intents of 'Forging process design'. The intent context graph for this intent is shown in Fig. 5. From this

graph and the information available in the problem context, the system constructs the query graph shown in Fig. 6. This query graph is matched against the context graphs of knowledge elements stored in the repository. Matching is based on a distance measure that takes into account class similarity, property similarity and related object similarity (recursively) [7]. Matching also takes into account the relative weights of entities in the query graph. A detailed description of the matching algorithm is beyond the scope of this paper.

The query graph of Fig. 6 matches the context graph of the knowledge element 'Billet Volume Estimation Procedure' shown in Fig. 8. Note that gear geometry matches Axisymmetric geometry since ontological reasoning infers that the latter subsumes the former.

V. RELATED WORK

CommonKADS [8] is a well-known method that proposes a model-based approach to knowledge engineering. However, CommonKADS models are general purpose in nature and are meant to provide a broad roadmap. In contrast, we propose a knowledge modeling framework that is specifically designed for the needs of an engineering design domain that supports integrated design across a number of subject areas, with an open ended architecture for adding new design domains. To support this, we propose a meta-modeling approach to model ontologies, intents, contexts and knowledge elements.

Knowledge Based Engineering (KBE) [9, 10] is another methodology with its roots in computer-aided design (CAD) and product lifecycle management (PLM) systems. It facilitates the creation of a product design based on best practices, geometry and data related to a product family. However, it does not provide a framework for modeling knowledge spanning multiple problem dimensions such as material, product and manufacturing process. Neither does it provide a means to process knowledge in a context-sensitive manner.

Expert systems have long been used for building knowledge bases. While expert systems support automated inference, not all knowledge is amenable to formal representation and reasoning. We need a system that supports both formal and informal knowledge representations (such as documents), and a means to reason about the knowledge itself so as to be able to deploy right knowledge in right problem contexts.

The role of context in reasoning has been extensively studied in the AI literature [11, 12, 13]. Context also plays a big role in knowledge retrieval. Knowledge is generated in a certain context. At the same time, knowledge to serve a given intent varies from problem context to problem context. It is essential to match these contexts to retrieve right knowledge. We propose an ontological approach to describe contexts, and an ontology-driven context-matching method to retrieve knowledge. Moreover parts of a context can be specified at the meta-level, thereby generalizing the context.

Case-based reasoning (CBR) provides a means to retrieve cases based on problem context matching. There are also approaches that marry CBR with ontologies [7], where both

problems and queries are described in terms of ontologies. This is somewhat similar to our approach where we also describe knowledge element contexts and query graphs in terms of ontologies. However, the key difference is the intent model, using which we can automatically generate context-appropriate knowledge queries.

VI. SUMMARY/CONCLUSIONS

In this paper, we have given an overview of a platform for integrated computational materials engineering. We have discussed its knowledge engineering requirements and proposed a knowledge modeling framework to address these requirements. We have discussed a meta-modeling framework for flexible ontology modeling. We have discussed a knowledge representation model where intents can be modelled and ontological contexts of knowledge elements can be captured. We have also discussed how these models can be used for context-sensitive knowledge retrieval.

REFERENCES

- [1] NRC Report (2008) Integrated Computational Materials Engineering: A Transformational Discipline for Improved Competitiveness and National Security. The National Academies Press, National Research Council, Washington, D.C.
- [2] TMS Study Report on Integrated Computational Materials Engineering (ICME) – Implementing ICME in the Aerospace, Automotive and Maritime Engineering, (2015) TMS, <http://www.tms.org/ICMEStudy>.
- [3] Konter, A.W.A., Farivar, H., Post, J. and Prah, U. Industrial Needs for ICME. JOM: the journal of the Minerals, Metals & Materials Society, 2015.
- [4] Gautham, B.P., Singh, A.K., Ghaisas, S.S., Reddy, S. S. and Mistree, F. (2013a) PREMAP: A Platform for the Realization of Engineered Materials and Products, A. Chakrabarti and R. V. Prakash (eds.), ICORD'13, Lecture Notes in Mechanical Engineering, Springer India, pp. 1301-1313.
- [5] Kulkarni, V., Reddy, S., & Rajhooj, A. (2010). Scaling up model driven engineering—experience and lessons learnt. In Model Driven Engineering Languages and Systems (pp. 331-345). Springer Berlin Heidelberg.
- [6] Baader F., Calvanese D., et al. The description logic handbook: theory, implementation, and applications. Cambridge University Press.
- [7] Ralph B., Martin S. Case-Based Reasoning and Ontology-Based Knowledge Management: A Perfect Match?. Journal of Universal Computer Science. vol. 9, no. 7 (2003), 608-626.
- [8] Guus T.S., Hans A. Knowledge engineering and management: the CommonKADS methodology. MIT Press Cambridge, MA, USA.
- [9] Verhagen, W. J., Bermell-Garcia, P., van Dijk, R. E., & Curran, R. (2012). A critical review of Knowledge-Based Engineering: An identification of research challenges. Advanced Engineering Informatics, 26(1), 5-15.
- [10] La Rocca, Gianfranco. "Knowledge based engineering: Between AI and CAD. Review of a language based technology to support engineering design." Advanced engineering informatics 26.2 (2012): 159-179.
- [11] Öztürk, Pinar, and Agnar Aamodt. "A context model for knowledge-intensive case-based reasoning." International Journal of Human-Computer Studies 48.3 (1998): 331-355.
- [12] Montani, Stefania. "How to use contextual knowledge in medical case-based reasoning systems: A survey on very recent trends." Artificial intelligence in medicine 51.2 (2011): 125-131.
- [13] Brézillon, Patrick. "Context in problem solving: a survey." The Knowledge Engineering Review 14.01 (1999): 47-80.
- [14] MOF Model to Text Transformation Language. www.omg.org/spec/MOFM2T/1.0

Building Ontology for Different Emotional Contexts and Multilingual Environment in Opinion Mining

Wan Tao, Tao Liu, Wei Yu, Gan Huang

School of Computer and Information Science, Anhui Polytechnic University
Key Laboratory of Computer Application Technology, Anhui Polytechnic University
Wuhu, China
{Taowan, Liutao}@ahpu.edu.cn

Abstract—With the explosive growth of various social media applications, individuals and organizations are increasingly using their contents (e.g., reviews, forum discussions, blogs, micro-blogs, comments, and postings in social network sites) for decision making. These contents are typical big data. Opinion mining or sentiment analysis focuses on how to extract emotional semantics from these big data to help users to get a better decision. That is not an easy task because it faces many problems, such as different context may make the meaning of the same word change variously, at the same time multilingual environment restricts the full use of the analysis results. Ontology provides knowledge about specific domains that are understandable by both the computers and developers. Building ontology is mainly a useful first step in providing and formalizing the semantics of information representation. We proposed an ontology DEMLOnto based on six basic emotions to help users to share existed information. The ontology DEMLOnto would help in identifying the opinion features associated with the contextual environment, which may change along with applications. We built the ontology according to ontology engineering. It was developed on the platform Protégé by using OWL.

Keywords—Ontology; opinion mining; social media; emotional context; multilingual environment; OWL

I. INTRODUCTION

Today large amount of information is supplied by the social media users. They share information about products, services and their experiences on the social networks. The information may contain users' sentiments, such as joy, sad, like, or dislike (Sentiment and emotion mean similar, in the following, we usually use emotion in a more specific situation). Users' sentiment orientation and emotion of the topic or event can not only provide decision-making basis in business but also provide support for government's public opinion monitoring. But it is very hard to integrate users' comments and feedback into normal application system. Currently, opinion mining is increasingly important than ever before, especially in customer preference analysis and prediction. Most opinion mining attempt to identify the polarity of sentiment in three categories: positive, negative or neutral. But how to identify the polarity of sentiment is a difficult task. The same word in different contexts may convey different emotions. For example, the word 'high' in sentence 'My salary is high' represents the positive emotion but in 'The price is high' may represent the negative emotion. Otherwise,

multilingual applications have produced many problems, such as how to map a word in Chinese to a similar word in English.

In this paper, we construct ontology *DEMLOnto* to help to solve the above problems. This ontology would help in identifying the opinion features associated with the different contextual environment that contains positive or negative sentiments. Also some methods are proposed to deal with multilingual environment. The remaining portion of the paper is organized as follows: Section II firstly introduce the special environment of social media, then describes emotion and opinion mining, provides the background research used for why and how to integrate the ontology into opinion mining; Section III provides the definition of our ontology *DEMLOnto*, and focuses on how to design our ontology on emotions and multi-languages these two aspects; Section IV provides constructs of the ontology in OWL(a Web Ontology Language), focuses on discussion about how to design ontology, explain how to get new describing construct, provides the building process on the Protégé and introduce how to use *DEMLOnto* to extract information from the social media; and Section V briefs on conclusions and future work.

II. BACKGROUND RESEARCH

A. Social Media Environment

Social media supports the interaction among people in which they create, share, discuss, or exchange ideas in virtual communities and networks. Sentiment expression in micro-blog posts often reflects user's specific individuality due to different language habit, personal character, opinion bias and so on [1].

Social media brings us with typical big data. Firstly, it produces large amount of data. For instance, daily online QQ users count for 160 million. Their storage is 300G. Secondly, data update fast. Twitter posts about 14300 per second. Thirdly, social media data is extremely various. This kind of variety is not only limited into its data type which is structured or mainly unstructured, but also means the following: many new words produced by media users (e.g. “童鞋” means “同学”, which is “classmate” in English); kinds of contents inserted by supplying simple “Like” or “Dislike” tools, star-rating systems, tag-based annotation and navigation, and so forth. Lastly, there is uncertainty in social media data. While using the same word, people may deliver different sentiment

orientations depending on the underlying context, just as our example in Section 1. Our work mainly has relationship with the last two points. We found that vocabulary is the basis of sentiment analysis. When we do sentiment analysis, first of all, we should have a reserve of lexical knowledge, so that the massive sentiment analysis does not require relearning subjective point of view, then the efficiency of sentiment analysis is improved greatly. Ontology knowledge base is a good vocabulary knowledge base, which can help to understand concepts and the relationship among all the attributes in the concepts.

B. How to do Opinion Mining

Opinion mining, also called sentiment analysis, is the field of study that analyzes people’s opinions, sentiments, evaluations, appraisals, attitudes, and emotions towards entities such as products, services, organizations, individuals, issues, events, topics, and their attributes[2]. The task of determining positive and negative orientations of the information present in textual form is considered as a fundamental issue in opinion mining.

In general, opinion mining has been investigated mainly at three levels: document level, sentence level, entity and aspect level. It can be used for three varied objectives: polarity identification, subjectivity or objectivity identification and feature/aspect based analysis[2]. In this paper we focuses on opinions which express or imply positive or negative sentiments then make a summary at different levels.

Opinion mining consists of the following tasks[3]: 1) entity extraction and categorization which is often done by ETL technology; 2) aspect extraction and categorization which mainly focus on opinion features; 3) opinion holder extraction and categorization; 4) time extraction and standardization; 5) aspect sentiment classification which often determine the actual feeling about the mining object by assign a numeric sentiment rating to the aspect; 6) outputting the analysis results in tuples.

C. Opinion and Emotion Description

According to [3], an opinion can be defined as a quintuple

$$(e_i, a_{ij}, s_{ijkl}, h_k, t_l),$$

where e_i is the name of an entity, a_{ij} is an aspect of e_i , s_{ijkl} is the emotional on aspect a_{ij} of entity e_i , h_k is the opinion holder, and t_l is the time when the opinion is expressed by h_k . The emotional s_{ijkl} is positive, negative, or neutral, or expressed with different strength/intensity levels, e.g., 1 to 5 stars as used by most review sits on the Web. That is, the opinion s_{ijkl} must be given by opinion holder h_k about aspect a_{ij} of entity e_i at time t_l . Any mismatch is an error. At last, the feature pairs and emotional words can help us make decision.

Emotions are our subjective feelings and thoughts. Based on [4], people have six primary emotions, i.e., love, joy, surprise, anger, sadness, and fear, which can be sub-divided into many secondary and tertiary emotions. Each emotion can also have different intensities. The strength of an emotion or opinion is typically linked to the intensity of certain emotions, e.g., joy and anger.

We think if we can integrate features (i.e. emotions, aspects, etc) into the ontology, we will share them more comfortable. In Section 3 we will focus on emotional context, polarity strength and multi-language these aspects by integrating them into ontology.

D. Ontology Description

Ontology has been defined as the specialization of the conceptualization by [5]. The main aim of ontology is to provide knowledge about specific domains that are understandable by both the computers and developers. Ontology plays important roles in sharing sources and defining terms precisely for future uses such as meta-data. It also helps to interpret a text review at a finger granularity with shared meanings and provides a sound semantic ground of machine understandable description of digital content.

In former researches, reference [6] has proposed a feature ontology that uses a multidimensional model to integrate customer’s characteristics and their comments about products. This approach first identifies the entities and then emotions present in the customers reviews related to mobiles are transformed into an attribute table by using a 7 point polarity system (-3 to 3). The limitation of their approach is that the ontology proposed by them is too general and is lack of reasoning ability among multiple products. In [7], a whole framework for building domain ontology of video is proposed. The semantic link network model is used to mine and organize video resources based on their associations. A semantic-based video organizing platform is provided for searching videos.

However, we found that even previous researches have focused emotion analysis at different levels but domain knowledge, context relationship, and multilingual application were not considered together during those researches. We think if ontology can be integrated into the 6 tasks of opinion mining (introduced in Section 2), it will make the mining results to be structured, make the results easy to be utilized and shared.

III. THE ONTOLOGY OF EMOTIONS AND MULTI-LANGUAGE

A. How to Deal with Emotions

We have developed *DEMLOnto* ontology ascended from [8,9] with some changes. Our ontology is designed on six basic emotions: *love, joy, anger, sad, fear, surprise*. They are structured in a taxonomy that covers from six basic emotions concepts to the most specific emotional words as data objects or instances. Each of the emotional data is related with the seven emotional strength levels by means of data ranges, as well as three polarity values to stand for positive, negative, or neutral.

Definition 1: Ontology is defined as

$$O = \{C, A^C, R, A^R, H, I, X\},$$

where C is a set of concepts; A^C is a set of multiply properties which one is belong to a concept; R is a set of relationships; A^R is a set of multiply relationships which one is belong to a property; H is a set of hierarchy relationships among concepts; I is a set of instances; X is a set of axioms.

In Fig. 1, we explain how concept Joy and Anger are constructed into the emotional ontology. In particular, when the application environment is not the same, we will make record of the different emotion-feature pairs into the ontology base, such as (price, i-Phone, highness, -1), (Tom's salary, incoming, highness, +2).

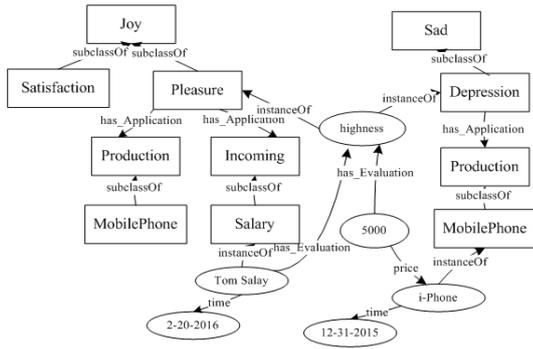


Fig. 1 Emotional Ontology(Fragment)

Fig.2 is a fragment of our ontology *DEMLOnto*'s definition. In definition, we use many properties to explain the feature of an emotion. For instance, polarity of an emotion which value is [1, 0, -1] respectively represents the emotion is positive, negative, or neutral; power represents the degree of positive or negative. We use (-3, 3), there are totally 7 numbers to stand for the different degree of an emotion. Why do we design two properties to measure an emotion? We do it just for that different applications maybe require different granularity in analysis process.

```

 $O_{em} = \{ C_{em}, A_{em}^C, R_{em}, A_{em}^R, H_{em}, I_{em}^C, X_{em} \}$  where
 $C_{em} = \{ Joy, Anger, Sad, Fear, Surprise, Love \}$ 
 $A_{em}^C = \{ A_{em}^C(Joy), A_{em}^C(Anger), \dots \}$ 
 $A_{em}^C(Joy) = \{ polarity(integer), power(integer) \}$ 
 $R_{em}^C(Joy) = \{ subclassOf, instanceOf, is\_Similarwith, has\_Application, has\_Apptime, has\_Evaluation, \dots \}$ 
 $A_{em}^R(Joy, is\_Similarwith) = \{ inChinese(text), inEnglish(text), inSpanish(text), \dots \}$ 
 $H_{em} = \{ Is-a, Superclass, Subclass, \dots \}$ 
 $I_{em}^{pleasure} = \{ highness, goodness, fineness, \dots \}$ 
 $X_{em} = \{ polarity\_rules, power\_rules, \dots \}$ 

```

Fig. 2 Ontology DEMLOnto Definition (Fragment)

B. How to Deal with Multi-Languages

In order to deal with multiply languages in some application, we extended the original ontology with some changes from [8].

The ontology has three root classes: *Emotion*, *Word*, *Application*. *Emotion* is the root for all the emotional concepts. *Word* is the root for the emotion-denoting words, i.e. the words which each language provides for denoting emotions, easily to be extended. Now we supposed it originally had two subclasses: *EnglishWord* and *ChineseWord*. All original words can get by information extraction from the Web (just as task 1 of opinion mining). *Application* includes objects which the emotions come from.

And in Fig.2 we designed *is_Similarwith* relationship to help us dynamically classify the new words which are

provided for denoting emotions into *Word* concept. The ontology's structure will be listed in Fig. 4 which is worked out by Protégé in Section 5.

IV. BUILDING THE ONTOLOGY IN OWL

A. OWL Overview

OWL is a recommendation of the W3C from 2004. In 2009, OWL 2 is proposed as an extension and revision of the OWL. OWL 2 also has three increasingly-expressive sublanguages: OWL Lite, OWL DL, and OWL Full. Just as the former OWL. OWL Lite is simple but lack of logical expression ability; OWL Full is powerful for representing complex statements but not useful for reasoning with them due to their computational properties. OWL DL is the subset of OWL designed for applications that need the maximum expressiveness without losing computational completeness and decidability. It is based on Description Logics, a particular fragment of first order logic, in which concepts, roles, individuals and axioms that relate them are defined.

In our work, we apply OWL DL to construct our ontology.

B. Building Process of DEMLOnto

According to [10], building an ontology includes: 1) defining classes in the ontology; 2) arranging the classes in a taxonomic (subclass-superclass) hierarchy; 3) defining properties and describing allowed values for these properties; 4) filling in the values for properties for instances.

At first, English emotional classes were extracted as the opinion as positive and negative based on the scales developed in previous works by [6], and Chinese classes were gained from [11]. All of these are read and evaluated by human beings.

We use Protégé 5.0 as the building platform. Protégé is mainly an ontology editor developed by Stanford University. Its reasoners can automatically classify the concepts into a hierarchy of emotional concepts.

In OWL, we can define new constructs and axioms to fit for our needs. For instance, in Section 3 we have designed *is_Similarwith* to handle similar words in different language contexts. It helps us map the Chinese words to the similar English words, and then they can be classified into the correct emotional class by the reasoner. The results are shown in Fig. 3. We can see in the Fig. 3(a), “狂怒”(which means ‘Fury’) is a subclass of ChineseWord. At the same time “狂怒” has property *is_Similarwith* with “Fury” that is a subclass of Anger. As seen in Fig. 3(b), “狂怒” is automatically add into subclasses of Anger by the reasoner Pellet(a plug-in of Protégé). Otherwise, we also defined *has_application*, *has_apptime*, *has_evaluation*, etc properties in ObjectProperty of different concepts. The ontology structure is shown in Fig.4, which is displayed by OntoGraf (a plug-in of Protégé).

In summary, the ontology *DEMLOnto* can be supplementing accumulatively under the help of opinion extraction. Conversely, the built ontology can fix the mining results into common knowledge to help us eliminate ambiguity in opinion mining.

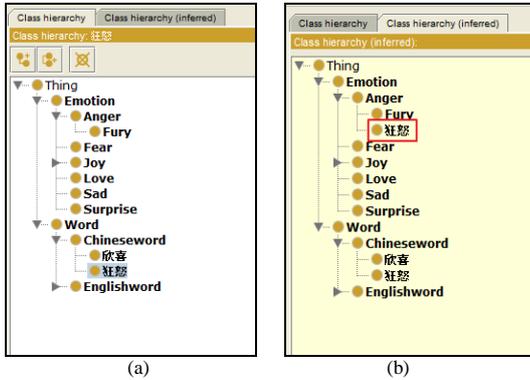


Fig. 3 Reasoning in Multi-Language

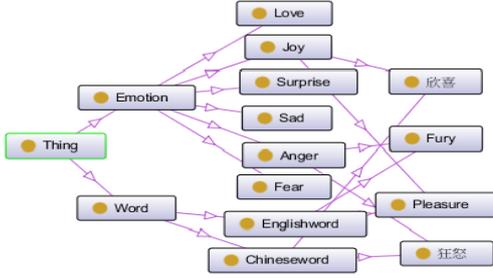


Fig. 4 DEMLOnto Structure (Fragment)

C. How to Use DEMLOnto

Ontology can help us to extract effective opinion, to make us mine more useful results. Now, we have used *DEMLOnto* to help us to extract opinion from micro-blog on domains of weather, book, and shopping. Under the guidance of the emotion ontology, we annotate entities at DOM level based on context distance and co-occurrence number. The annotation results were added to the entity record. And similarity of the ontology and entity is calculated by using (1).

$$Sim(C, O) = \frac{\max \sum_{j=1}^n ReI(C, A_j)}{\sqrt{mn}} \quad (1)$$

$$ReI(C, A_j) = \alpha \times \left(\frac{Itemspan(C, A_j)}{\max \{Itemspan(C, A_j)\}} \right) + (1 - \alpha) \times \frac{Itemtimes(C, A_j)}{\sum_{j=1}^n Itemtimes(C, A_j)}$$

$$Itemspan(C, A_j) = \frac{\sum_{i=1}^m Span(c_i, A_j)}{m}$$

where O is ontology *DEMLOnto*; m is properties of an application entity in *DEMLOnto*; n is properties of an entity in web page being extracted; C represents an entity semantic values set; $Span(c_i, A_j)$ is context distance between semantic C_i and ontology attribute A_j in the same page; $Itemspan(C, A_j)$ is mean value of context distance between each semantic item and the attribute A_j in the same page of an entity C .

According to the Google ranking, we selected five top social media sites which mainly about books, weather and shopping to our research. If the computation output of (1) is large than 0.62(this value is determined by several tests), we judge that it is a good result. The evaluation index of information extraction is the accuracy rate of P, the recall rate

of R and F-measure. The experiment result is shown in TABLE I.

TABLE I. THE COMPARISON OF DIFFERENT DOMAIN

Domain	P(%)	R(%)	F(%)
Weather	84.8	84.4	84.6
Book	83.9	82.8	83.3
Shopping	82.4	81.7	82.1

V. CONCLUSION

Currently, our ontology is developed based on manually selected classes from some baseline datasets, and our research work is limited at extraction stage. Our next target is to make use of machine learning method to develop ontology from actual applications automatically and effectively. Also we will integrate ontology into opinion mining deeply. ProtégéOLV is a plug-in published in 2015. It can help us searching for relevant vocabularies in the Linked Open Data catalogue, for easing the development of ontology by reusing existing vocabularies at low fine grained level. So, taking use of ProtégéOLV may largely promote the efficiency of building and reusing of the ontology. In the next, we will focus on it.

Acknowledgment

This research is supported by National Natural Science Foundation of China (61300170), by Key Laboratory of Computer Application Technology of Anhui Polytechnic University (JSJKF201507).

References

- [1] Wu F., Song Y., Huang Y., Microblog sentiment classification with contextual knowledge regularization. Twenty-Ninth AAAI Conference on Artificial Intelligence 2015.
- [2] Dinakar S, Andhale P, Rege M. Sentiment analysis of social network content. Information Reuse and Integration (IRI), 2015 IEEE International Conference on IEEE, 2015:189-192.
- [3] Liu B, Zhang L. A survey of opinion mining and sentiment analysis, Mining Text Data. Springer US, 2012:459-526.
- [4] Parrott, W. Gerrod. Emotions in social psychology, Psychology Press, 2001.
- [5] T. R. Gruber. A translation approach to portable Ontology Specifications. Knowledge Acquisition, 1993,5(2):199-220.
- [6] Yaakub M R, Li Y, Zhang J. Integration of sentiment analysis into customer relational model: The Importance of Feature Ontology and Synonym. Procedia Technology, 2013, 11(1):495-501.
- [7] Zheng Xu, Yunhai Liu, Lin Mei, et al. Semantic based representing and organizing surveillance big data using video structural description technology. The Journal of Systems and Software, 2015, 102:217-225.
- [8] Baldoni M, Baroglio C, Rena P, et al. From tags to emotions: Ontology-driven sentiment analysis in the social semantic web. Intelligenza Artificiale, 2012, 6(1):41-54.
- [9] Francisco V, Gervás P, Peinado F. Ontological reasoning to configure emotional voice synthesis. Proceedings of the 1st international conference on Web reasoning and rule systems. Springer-Verlag, 2007:88-102.
- [10] Noy N F, McGuinness D L. Ontology Development 101: A Guide to Creating Your First Ontology. And Stanford Medical Informatics, 2001.
- [11] Xu Linhong, Lin Hongfei, et al. Constructing the affective lexicon Ontology, Journal of the China Society for Scientific and Technical Information. 2008, 27(2):180-185

Building the Multi-layer Theory of Association Semantic based on the Power-law Distribution of Linking Keywords

Guangli Zhu, Xiaojun Tang, Shunxiang Zhang
School of Computer Science and Engineering,
Anhui University of Science & Technology
Huainan, China
glzhu@aust.edu.cn, 1262989796@qq.com,
sxzhang@aust.edu.cn

Zheng Xu*
The Third Research Institute
the Ministry of Public Security
Shanghai, China
xuzheng@shu.edu.cn
*Corresponding author

Abstract—Web information contain plentiful, significant knowledge which is eager to be explored by users. Effective semantic layered technology not only can provide theoretical support for knowledge discovery in Web resources, but also can improve the searching efficiency of the related information system. This paper builds the multi-layer theory of association semantic based on the power-law distribution of linking keywords. First, some experiments of four types of keywords with different linking role are done to discover the possible distribution law. Experiment results show that four types of keywords are all reveal power-law distribution. Then, based on the discovered power-law distribution, the multi-layer theory of association semantic is built. The multi-layer theory of association semantic can provide a theoretical support for knowledge recommendation with different particle size on Association Link Network (ALN).

Keywords- Association Link Network, power-law distribution, multi-layer theory of association semantic, knowledge discovery in Web resources.

I. INTRODUCTION

Web information contain plentiful, significant knowledge including explicit and implicit knowledge [1]. How to organize these Web information for facilitating knowledge discovery has been deeply by some researchers. Association Link Network (ALN) is a kind of Semantic Link Network built by mining the association relations among Web resources for effectively supporting Web intelligent application such as Web semantic association search, Web knowledge discovery and recommendation[2,3]. Xu et al. have studied on cloud environment for surveillance data management using video structural description[4], generating temporal semantic context of concepts [5]. Zhu et al presents discovering and learning communities and emerging semantics in Semantic Link Network [6]. With the rapid development of information technology, human kinds are more likely to read and share information by similar intelligent applications. For example, the distributed and collaborative learning[7], semantic representation of scientific documents for supporting e-learning[8], discovering and searching of correlation between

shared resources[9], and smart component technologies for human centric computing [10],etc.

Based on the these research about ALN, this paper explores the multi-layer theory of association semantic to provide theoretical support of knowledge discovery, recommendation on different particles/layers for users. The significant contributions of this paper are as follows:

- The power-law distribution of four types of keywords with different linking role. According to the role of association semantic link, two kinds of semantic features are defined. Further, association semantic links are with extracted on different supports to compute the distribution of four kinds of keywords. All the keywords with association semantic role reveal obvious power-law distribution characteristic.
- The multi-layer theory of association semantic. Based on the power-law distribution of association keywords, Association Semantic Concentric (ASC) is defined. And corollary about the relations between the ASC and the exponent of power-law distribution is presented. Larger exponent of power-law distribution leads to smaller average value of ASC. In addition, the corollary about changing trend of ASC among any two adjacent layers is proved.

To the best of our knowledge, the multi-layer method of association semantic has not been well studied in existing work.

The rest of this paper is organized as follows. In section II, two semantic features and four kinds of keywords in association links are given. In section III, The distribution of four kinds of keywords is presented. In section IV, the layered theory of association semantic are proposed. In section V, the multi-layer theory is validated by experiments. Finally, section VI concludes the work of this paper.

II. TWO SEMANTIC FEATURES AND FOUR KINDS OF KEYWORDS IN ASSOCIATION LINKS

Different types of keywords have different roles in association semantic links. In this section, we will analyze the

semantic association tendency (i.e., active traction and passive traction) of keywords to divide these keywords into four types of keywords. Note that these keywords are domain keywords which are extracted by the prior extracting method from Web resources[2].

A. Two semantic features

Definition 1: Active Traction Feature of Keyword (ATF)

Active Traction Feature of Keyword (ATF) is a kind of semantic feature which is owned by antecedent keyword in a keyword-level association semantic link. In our research, a keyword with active traction feature must satisfy the following two conditions.

- A keyword k_m with ATF must be extracted from a domain Web resources on a given time window. This can ensure that this keyword can be used as a element of representing domain knowledge.
- The keyword must be used as an antecedent keyword occurred in one or more association semantic links.

This type of semantic feature comes from two types of knowledge. One is well-known knowledge which is human’s association cognitive sense hidden in human mind. For example, association link {“insulin”->“diabetes”} in health domain belongs to this type of knowledge. The other is unknown knowledge which must be mined from massive data. For example, association link {“polio”->“Bill Gates”} belongs to this type of knowledge. Here, “insulin” and “polio” all have active traction semantic feature.

Usually, a keyword with ATF can be used to describe the attribute of an object, event. Or it is used to describe the part of an object, event. For example, in Figure 1 (a), the keyword “woman” is an object which has been listed four attributes (described using keywords with active traction feature) such as “supplement”, “menopause”, “pregnancy” and “fertility”. In Figure 1 (b), “Google” and “apple” are two the described objects, which have been listed some active traction features such as “search”, “iPad” and so on.

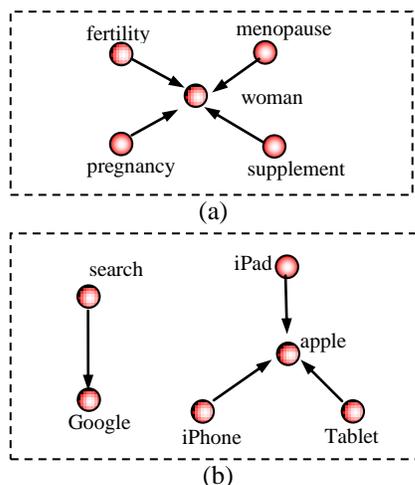


Figure 1. Two semantic features in association semantic link

Definition 2: Passive Traction Feature of Keyword (PTF)

Passive Traction Feature of Keyword (PTF) is another kind of semantic feature which is owned by descendant keyword in a keyword-level association semantic link. In our research, a keyword with passive traction feature must satisfy the following two conditions.

- A keyword k_m with PTF must be extracted from a domain Web resources on a given time window. This kind of keyword is also used as a element of representing domain knowledge.
- The keyword must be used as an descendant keyword occurred in one or more association semantic links.

Usually, those concept keywords with high frequency has this type of semantic feature. For example, association link {“pregnancy”->“woman”} in human health domain (see Figure 1 (a)), “woman” has passive traction feature. Similar keyword example with PTF include “emission”, “climate” in environment domain, “Google” and “apple” in Internet domain.

In general, what semantic feature a keyword owns, ATF or PTF, is determined by its own semantic meaning. In anyway, it is the basic unit of semantic representation of Web resources. Semantic feature is the source of the semantic link from a Web resource to other Web resources. A Web resource is linked to other Web resources which are related to its ATF and PTF keywords. This problem will be discussed in the next section.

B. Four kinds of keywords

Based on the two basic semantic features, active traction and passive traction, all the extracted the keywords to be used as representation of domain knowledge from the Web resource [2] are divided into the following four types.

Definition 3: Active Traction Keyword (ATK)

For a keyword k_m , if it belongs to active traction keyword, it must satisfy the following two conditions.

- This keyword must be extracted from a domain Web resources and must be appointed/defined as one of the domain keywords by the TF/IDF method. This ensures that it can be used as a element of representing the semantic of Web resources.
- This keyword only has the semantic feature of active traction. That is, it must occur as the antecedent keyword of a keyword-level association semantic link on a given time window. Certainly, this association semantic link is the part of the keyword-level association semantic network.

Definition 4: Passive Traction Keyword (PTK)

Similar to the definition of ATK, if a keyword k_m belongs to passive traction keyword, it also must satisfy the following two conditions. One is that it must be one of the domain keywords extracted from a domain Web resources. The other is that it only has the semantic feature of passive traction. That is, it must occur as the descendent keyword of a keyword-level association semantic link on a given time window.

Definition 5: Bridging Traction Keyword (BTK)

As a Bridging Traction Keyword (BTK), it has two types of semantic feature. That is, in an association semantic link, it occurs as the antecedent keyword which has the semantic feature of active traction. In another association link, it appears as descendent keyword which has the semantic feature of passive traction. Certainly, it is a necessary condition that it is one of the domain keywords extracted from a domain Web resources. Obviously, this type of keywords has more important link role in the semantic representation of Web resources.

Definition 6: Non-Traction Keyword (NTK)

The fourth type of keyword is on the convert to the BTK in the semantic feature. It has not the two semantic features of active traction and passive traction. It is only used as one of the domain keywords extracted from a domain Web resources to represent the semantic of Web resources.

Based on the definitions of four kinds of keywords, we can simply plot their modes as Figure 2.

Usually, those concept keywords with high frequency has this type of semantic feature. For example, association link{"pregnancy"->"woman"} in human health domain (see Figure 1 (a)), "woman" has passive traction feature. Similar keyword example with PTF include "emission", "climate" in environment domain, "Google" and "apple" in Internet domain.

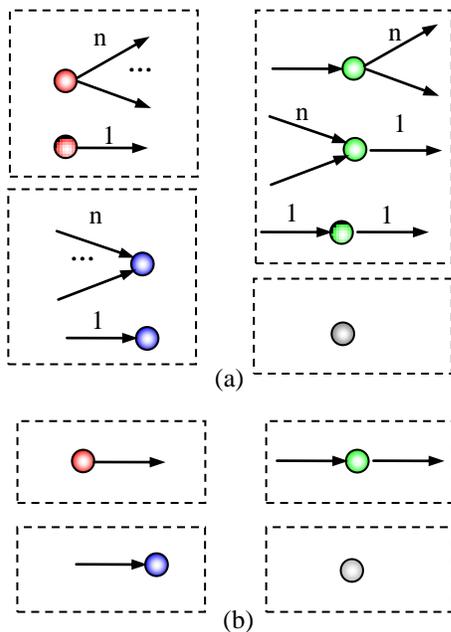


Figure 2. the graph of keywords set

III. THE DISTRIBUTION OF FOUR KINDS OF KEYWORDS

In this section, we use the Web resources of a month belonging to the health domain as the researched data set. We explore the distributions of four kinds of keywords on the different/variable supports of association semantic links (ASLs). At the same time, the distributions of all keywords and ASLs

are analyzed to do comparisons with the distributions of four kinds of keywords.

Our analysis is carried out on different supports. So, we first give the calculation procedure of the distribution by adjusting the support as following.

Algorithm 1: analyzing the distribution of four kinds of keywords and related ASLs

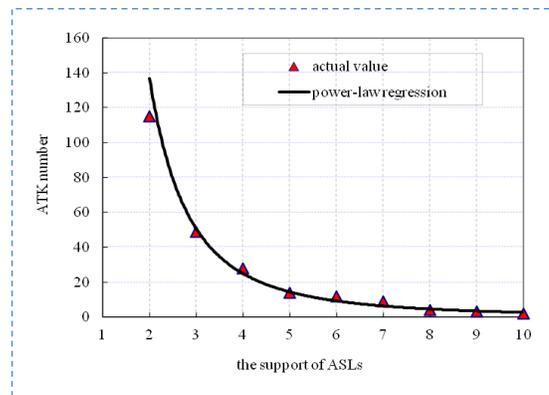
Input: the variable support, the semantic representation of Web resources on a month

Output: the number of different keywords and related ASLs

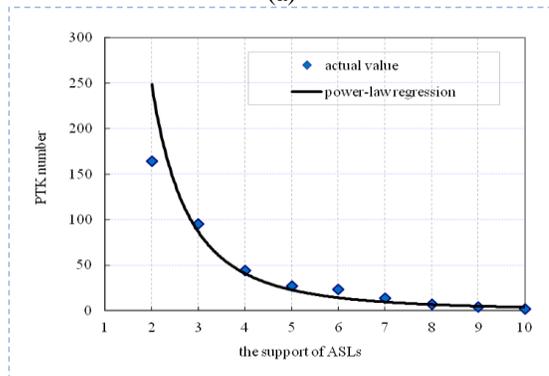
- 1: set the support of ASLs as 2
- 2: get the ASLs from the given semantic representation of Web resources by the method of ASL.
- 3: counting the number of four kinds of keywords occurring in these Web resources, and related ASLs;
- 4: if the number of the gotten ASLs is 0, then goto step 6;
- 5: else adding the support of ASL, and goto step 2;
- 6: end;

Note that, in this analyzing algorithm, the support is simplified as an integer variable. Usually, the support is the proportion of the document frequency and the total number of Web resources. Because the total number of Web resources is fixed in our adjusting process, so the support can be simplified as an integer variable.

According to this adjusting process, the distribution of four kinds of keywords, the related ASLs and their regression results are plotted as Figure 3.



(a)



(b)

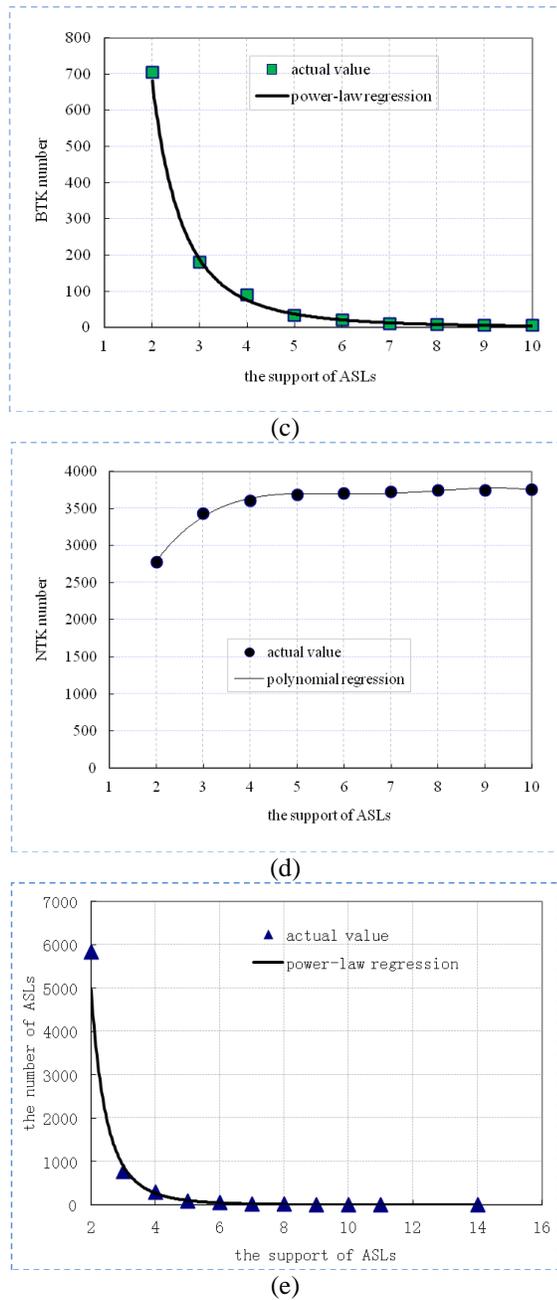


Figure 3. The distribution of four kinds of keywords and related ASLs

In our regression analysis, different regression methods are adopted. Based on the probable trend, we employ power-law regression to analyze the distributions of ATK, PTK, BTK and ASLs. The multinomial regression is selected to analyze the distribution of NTK. From Figure 3, we find that the distributions of ATK, PTK, BTK and ASLs follow power-law function. And the distribution of NTK follows the four times multinomial function.

Further, the Web resources of Internet and environment domain are selected to analyze the distribution of four kinds of keywords and related ASLs. The achieved parameters of regression analysis are listed in Table I.

TABLE I. THE POWER-LAW EXPONENT AND ITS COMPLEX CORRELATION COEFFICIENT OF REGRESSION ANALYSIS ON FOUR DIFFERENT SUPPORT

		ATK	PTK	BTK	ASL	NTK
H	b	2.25	2.87	2.39	3.76	/
	R	0.98	0.99	0.96	0.99	0.99
I	b	2.71	3.47	2.77	4.10	/
	R	0.97	0.987	0.94	0.99	0.98
E	b	2.19	3.11	2.95	4.21	/
	R	0.98	0.97	0.96	0.99	0.99

In Table I, “ b ” denotes the exponent of power-law regression, “ R ” denotes the the complex correlation coefficient of regression analysis. From Table I, we can find the following coincident result.

The result of regression analysis: four kinds of keywords and the related ASLs except NTK strictly follow power-law distribution. Higher complex correlation coefficients show the correctness of regression analysis. More importantly, the “core semantic” gradually occurs with the bigger support of ASLs.

IV. THE LAYERED THEORY OF ASSOCIATION SEMANTIC

In this section, we first give the basic idea of the layered theory. Then, the layered theory based on semantic concentric degree is presented.

A. The basic idea of the layered theory

From the result of regression analysis in section 3, all the keywords with semantic traction feature, ATK, PTK and BTK, follow power-law distribution. At the same time, the keywords and related ASLs are becoming less with the occurring of larger support of ASLs. This means that some “basic semantic” are stripped and the “core semantic” gradually occurs.

If we regard the keywords-level association semantic network at a given support as a filled circle. Then we have some concentric circles on different supports(from Figure 4(a) to Figure 4(b)). Further, these concentric circles can be simplified/reduced as Figure 4(c). After that, the simplified concentric circles can be mapped into multi-layer semantic shown as Figure 4(d).

Definition 7: Three-layer Association Semantic(TL-AS)

For the Web resources on a given time window, we can get some keywords-level association semantic network (k-ALN) G_1, G_2, \dots, G_m (corresponding to some concentric circles) on different supports $\{fre_1, fre_2, \dots, fre_m\}$. If we can simplified them as Figure 4(c). Then, three-layer association semantic can defined as the following.

- G_1 can be named as “basic semantic” of the Web resources on the given time window. This corresponds to the outer of concentric circles in Figure 3(c).
- G_i can be named as “main semantic” of the Web resources on the given time window. This corresponds to the middle of concentric circles in Figure 3(c).

- G_j can be named as “core semantic” of the Web resources on the given time window. This corresponds to the inner of concentric circles in Figure 3(c).

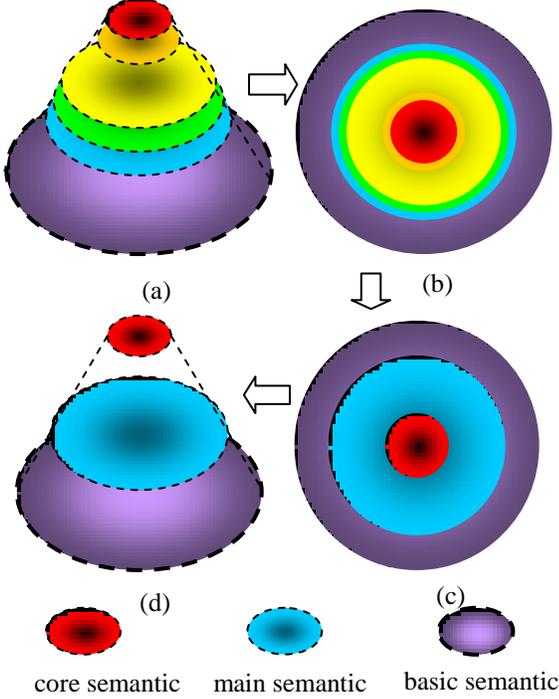


Figure 4. The basic idea of the layered theory

B. The layered theory based on semantic concentric degree

Based on the basic idea of the layered theory, we present the detail layered theory and its related concept.

Definition 8: Association Semantic Concentricity (ASC)

Association semantic concentric is the repeatability score of association semantic containing in two k-ALNs G_i, G_j . For the Web resources on a given time window, if two supports $\{fre_i < fre_j\}$ are given, two related k-ALNs G_i, G_j can be achieved by Luo’s method [3]. ASC $ASC(G_i, G_j)$ can be defined as,

$$ASC(G_i, G_j) = R_{G_j} / R_{G_i}, R_{G_i} = \sqrt{Count_i / \pi} \quad (1)$$

Where, R_{G_i} and R_{G_j} denote the semantic radius of G_i and G_j . $Count_i$ denotes the number of ASLs in G_i .

Corollary 1.the value field of ASC

For the Web resources on a given time window, G_i and G_j are two k-ALNs generated on different supports $fre_i < fre_j$, then we have, $0 < ASC(G_i, G_j) \leq 1$.

According to the Definition 7, for two supports fre_i and fre_j , if there is $fre_i < fre_j$. Then, based on generating method of k-ALNs, there be

$$G_j \subseteq G_i \quad (2)$$

Correspondingly, we have,

$$R_{G_j} \leq R_{G_i} \quad (3)$$

That is,

$$ASC(G_i, G_j) \leq 1 \quad (4)$$

In addition, for any an ALN, we have

$$R_{G_j} > 0 \quad (5)$$

So,

$$ASC(G_i, G_j) > 0 \quad (6)$$

Combining formulas (4) and (6), we have,

$$0 < ASC(G_i, G_j) \leq 1 \quad (7)$$

Corollary 2. the relations between the ASC and the exponent of power-law distribution

Larger exponent of power-law distribution leads to smaller average value of ASC. It is true on the contrary.

For the Web resources on a given time window, we can set m supports fre , ALNs $G_1, G_2 \dots, G_m$, can be gotten. Larger exponent of power-law distribution means that the changing velocity of association semantic among these ALNs. This inevitably leads to smaller average value of ASC.

Corollary 3. the changing trend of ASC

For m ALNs $G_1, G_2 \dots, G_m$, if the number of their ASLs follow power-law distribution, then we have the sequence of ASC, $ASC(G_1, G_2), ASC(G_2, G_3) \dots, ASC(G_{m-1}, G_m)$, which is an increasing sequence.

Proof: Obviously, we only prove,

$$\forall i \in (1, m), ASC(G_{i-1}, G_i) < ASC(G_i, G_{i+1}).$$

According to the definition of ASC, we have,

$$\begin{aligned} ASC(G_{i-1}, G_i) &= R_{G_i} / R_{G_{i-1}} \\ &= \sqrt{a^* fre_i^{-b} / \pi} / \sqrt{a^* fre_{i-1}^{-b} / \pi} = [(i-1) / i]^{b/2} \end{aligned} \quad (8)$$

Similarly,

$$ASC(G_i, G_{i+1}) = [i / (i+1)]^{b/2} \quad (9)$$

Therefore,

$$ASC(G_{i-1}, G_i) / ASC(G_i, G_{i+1}) = [(i^2 - 1) / i^2]^{b/2} < 1 \quad (10)$$

That is,

$$ASC(G_{i-1}, G_i) < ASC(G_i, G_{i+1}) \quad (11)$$

Thus Corollary 3. is proved.

V. EXPERIMENT & RESULT ANALYSIS

In this section, we present the experimental data and result analysis to verify the multi-layer theory of association semantic.

A. Experimental data

Three domain news data are selected from Website <http://www.reuters.com>, including health, environment and internet, to building Web keywords-level ALN on different

supports. The time window of experimental data is set as a month.

For each domain news data, we independently execute the extracting method of association semantic link on a adjustable support. The initial value of the adjustable support is set as 2. Then, it is increased gradually. The increasing step is 1. The association semantic concentric of any two adjacent layer keywords-level ALNs are been computed by the definition 7. The computed results are listed in Table II.

B. The layered theory based on semantic concentric degree

In Table II, the column “*sup*” denotes two supports of two adjacent layers. “*H-ASC*”, “*I-ASC*” and “*E-ASC*” respectively denote the association semantic concentric of health, Internet and environment news domains. And the average value of association semantic concentric of each domain are listed in the last row of Table II.

From Table II, we can conclude the following conclusions:

- Larger exponent of power-law distribution leads to smaller average value of ASC. It verifies the correctness of the Corollary 2. We compare the results of power-law exponent listed in Table I and the average value of association semantic concentric listed in Table II. The keywords-level ALN of environment news domain has the largest power-law exponent and the smallest association semantic concentric.
- It has larger association semantic concentric at larger support between two adjacent layers. It verifies the correctness of the Corollary 3. From Table II, Although a few decreasing of association semantic concentric has occurred, this kind of increasing trend of two adjacent layers in total is obvious.

TABLE II. THE ASSOCIATION SEMANTIC CONCENTRIC OF THREE DOMAINS ON DIFFERENT SUPPORTS

<i>sup</i>	<i>H-ASC</i>	<i>I-ASC</i>	<i>E-ASC</i>
2,3	0.343132	0.414126	0.412968
3,4	0.637229	0.628666	0.474075
4,5	0.686762	0.639767	0.61808
5,6	0.758913	0.682575	0.641689
6,7	0.703211	0.662589	0.845154
7,8	0.788811	0.781736	0.894427
8,9	0.779194	0.797724	0.707107
9,10	0.8044	0.755929	无
aver	0.687706322	0.670389113	0.656214

VI. CONCLUSIONS

The multi-layer theory of association semantic not only can provide theoretical support for knowledge services such as knowledge discovery in Web resources, knowledge recommendation with different particle size but also can

improve the searching efficiency of searching system. Our contributions are as the follows:

(1) We have discovered the power-law distribution of four types of keywords with different linking role. According to the role of association semantic link, two kinds of semantic features are defined. Further, association semantic links are with extracted on different supports to compute the distribution of four kinds of keywords. All the keywords with association semantic role reveal obvious power-law distribution characteristic.

(2) We have built the multi-layer theory of association semantic. Based on the power-law distribution of association keywords, two corollaries are presented. One is the relations between the ASC and the exponent of power-law distribution. Larger exponent of power-law distribution leads to smaller average value of ASC. The other is about changing trend of ASC among any two adjacent layers.

ACKNOWLEDGMENT

This work was supported by the Natural Science Foundation of Anhui Province Universities (No. KJ2015A111), in part by the National Science and Technology Major Project under Grant 2013ZX01033002-003, in part by the National Science Foundation of China under Grant 61300202, and in part by the Science Foundation of Shanghai under Grant 13ZR1452900.

REFERENCES

- [1] S.X. Zhang, K. Lu, W. Liu, X. Yin, and G. Zhu, Generating associated knowledge flow in large-scale web pages based on user interaction. *Computer Systems Science and Engineering*, 30(5):377-389, 2015.
- [2] S.X. Zhang, X.F. Luo, J.Y. Xuan, et al., Discovering Small-World in Association Link Networks for Association Learning. *World Wide Web*, 17(2):229-254, 2014.
- [3] X.F. Luo, Zh. Xu, J. Yu, et al., Building association link network for semantic link on web resources. *IEEE Trans. Autom. Sci. Eng.* 8(3), 482-494, 2011.
- [4] Zh. Xu, et al, Semantic based representing and organizing surveillance big data using video structural description technology. *The Journal of Systems and Software*, 102, 217-225, 2015.
- [5] Zh. Xu, et al. Generating Temporal Semantic Context of Concepts Using Web Search Engines. *Journal of Network and Computer Applications*, 43:42-55, 2014.
- [6] H. Zhuge. Communities and Emerging Semantics in Semantic Link Network: Discovery and Learning. *IEEE Transactions Knowledge and Data Engineering*, 21(6), 785-799(2009)
- [7] Q. Li, R.W.H. Lau, T.K. Shih, et al., Technology supports for distributed and collaborative learning over the internet. *ACM Trans. on Internet Technology*, 8(2), 10:1-10:24, 2008.
- [8] X.F. Luo, N. Fang, et al., Semantic representation of scientific documents for the e-science Knowledge Grid. *Concurrency and Computation: Practice and Experience*, 20(7), 839—862, 2008.
- [9] N.Y. Yen, R.H. Huang, J.H. Ma, Q. Jin, and T. K. Shih, Intelligent route generation: discovery and search of correlation between shared resources. *Int. J. Commun. Syst.* 26, 732-746 (2013).
- [10] J. P. James, C. Antonio, H. Chang, K. Andrew, Introduction to the thematic issue on Ambient and Smart Component Technologies for Human Centric Computing. *JAISE* 6(1): 3-4, 2014.

Developer Recommendation with Awareness of Accuracy and Cost

*Jin Liu, Yiqiuzi Tian

State Key Lab of Software Engineering
Computer School, Wuhan University
Wuhan, China

*Corresponding author
jinliu@whu.edu.cn
tianyiqiuzi@whu.edu.cn

Liang Hong

School of Information Management
Wuhan University
Wuhan, China

hong@whu.edu.cn

Zhou Xu

State Key Lab of Software Engineering
Computer School, Wuhan University,
Wuhan, China

zhouxullx@whu.edu.cn

Abstract—As the scale and complexity of software products increase, software maintenance on bug resolution has become a challenging work. In the process of software implementation, developers often use bug reports, source code and change history to help solve bugs. However, hundreds of bug reports are being submitted every day. It is time-consuming and effortless for developers to review all the bug reports. To facilitate the assignment of bug reports, existing developer recommendation systems typically recommend the developer who has the fullest potential. However, bug reports are highly varied; time that the developers may spend fixing them is also important. To address the problem of developer recommendation, we propose a developer recommendation system with awareness of accuracy and cost (DRAC). This recommendation system is based on modern portfolio theory by striking a balance between accuracy and cost (time). We evaluate our approach with experiments on data collected from Bugzilla¹.

Keywords—Recommendation System; Portfolio Theory; Bug Triage; Accuracy and Cost

I. INTRODUCTION

Software development life cycle usually consists of six phases, i.e. requirements analysis, software design, implementation, testing, integration and maintenance. In the phase of implementation, many bugs appear with causes stemming from various stages. Previous studies have shown that more than 45% of modern software development is used to locate and fix bugs [1][2]. Generally, developers upload bug reports to bug-tracking platform, such as Bugzilla¹, Mantis², Trac³ and Redmine⁴ with a fixed format when they encounter bugs during implementation phase. It can also be considered as a crowdsourcing problem[3][4]. The bug reports are viewed and solved by the users of the platform, and it can be very time-consuming. Having a framework that makes predictions about the right developer with higher accuracy and less time is essential to shorten the lifecycle of the bugs [5]. Another fact about the bug report is that its volume is big and grows at a high rate at the same time. So strategy should be taken to response quickly when a new bug report is brought up [6].

¹ <http://www.bugzilla.org/>.

² <http://www.mantisbt.org/>.

³ <http://trac.edgewall.org/>.

⁴ <http://www.redmine.org/>.

This work is partly supported by the grants of National Natural Science Foundation of China (61572374, U1135005, 61303025).

DOI reference number: 10.18293/SEKE2016-125

Above all, the main problem in this area is the absent consideration of time cost. Existing works only do study on optimizing the possibility of finding the developer with the most similar history bug reports, but they fail to explore the cost. However, cost is very important, for it implies the time needed to get this bug fixed by a specific developer, and time cost has a great influence on software development process.

These observations lead to two goals: First, we need to formulate a recommendation framework to make predictions about developers with the consideration of both accuracy and cost. Second, to make quick response when a new bug report is brought up, we need to come up with a strategy managing the data efficiently.

In this paper, we propose a developer recommendation system with awareness of accuracy and cost (DRAC), which is modeled on the description of the bug reports. We divide the whole process into two phases - the offline learning stage and online recommendation stage. The first stage aims at learning the value of parameters; and the second stage uses the portfolio theory to make prediction about the best developer with a balance between accuracy and time cost. Finally, we evaluate our developer recommendation approach with experiments on a large-scale real-world dataset collected from project “Eclipse” on Bugzilla [7]. The experiment results validate the effectiveness and efficiency of our approach.

II. BACKGROUND

A. Data with Big Data Features

Big data has drawn huge attention from researchers recently [8]. Gartner listed big data as the second place in “Top 10 Critical Tech Trends for the Next Five Years” [9]. Technically speaking, big data is a collection of very huge dataset with a great diversity of types, so it becomes difficult to process by using state-of-the-art data processing approaches or traditional data processing platform. Big data has three features: variety, volume and velocity. Bug reports repository complies with these three specifications with its different-formatted components, large data scale and high growth rate. In Fig. 1, we can see that the size of bug report repository increases at a steady speed. At the end of the year 2015, the total amount of bug reports has passed 480,000, with nearly 50 reports brought up every day. It’s essential to reduce the volume of data before making recommendation. The applicable way includes feature selection and instance selection [10].

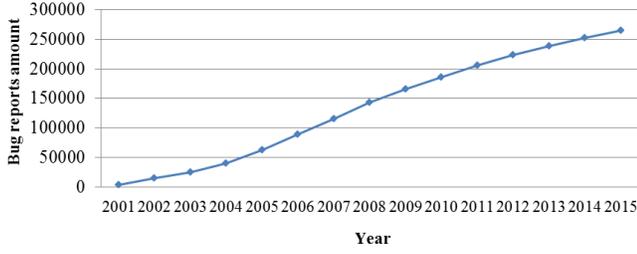


Figure 1. The increasing size of bug report repository

B. Recommendation System

Recommendation systems have become extremely common in recent years and are applied in a variety of applications. They typically produce a list of recommendations. Main methods in the area include content-based recommendation (CBR), collaborative filtering (CF) and association rule-based recommendation (ARBR). Each method adapts to different scenarios [11].

Considering our problem, the data source has two different forms [12]. The first is called metadata. It contains data like ID of the bug, the platform the bug belongs to, keywords and so on. The amount of metadata is fixed, and its value has a specific range. Using metadata to classify bug reports can be very hard, for that the information it implicates is very limited. The second is text data, which consist of description, summary and other developers' comments. With unlimited size and range, text data contain much richer information. So, different data sources should be weighted differently. In the field of developer recommendation, to accurately assign developers to bug reports, recent research treats it as a problem that optimizes recommendation accuracy and proposes solutions that are essentially instances of content-based recommendation (CBR). CBR mainly uses the content of the item and make predictions based on the similarity between contents. In this case, the data we use are the text content of bug reports.

In the recommendation system about bug reports, related researches aim at recommending right developers to bug reports which only consider the recommendation accuracy with old bug reports and the source code as data source. The main technology includes machine learning [13][14], information retrieval technique [14][15], tossing graph[16], fuzzy set[17] and Euclidean distance [18][19]. Compared to DRAC, this recommendation strategy has an obvious flaw. It only recommends the right developer who is capable of solving the problem, but doesn't consider the time the developer may spend on solving it. For urgent bug reports which have high severity level, it can be devastating to developing process.

C. Modern Portfolio Theory

Modern portfolio theory is originally proposed in the field of finance, which focus on the investment problem of financial market [20]. It's a mathematical framework for assembling a portfolio of assets such that the expected return is maximized for a given level of risk. Its key insight is that an asset's risk and return should not be assessed by itself, but by its contribution to a portfolio's overall risk and return. For example, an investor often wants to select a portfolio of n

stocks with a fixed investment budget, which will provide the maximum future return and the minimum risk. In our problem, the stocks can be regarded as bug reports, the future return and risk can be regarded as accuracy and cost of recommending bug reports.

In this problem, each bug report is taken as a recommender which recommends its own fixer to the considered bug report. When the modern portfolio theory is applied, risk vector and return risk are both needed, which in our case are cost vector and accuracy vector individually. By combining these two vectors, portfolio theory generates the weight of each bug report, i.e. the weight of the recommender.

III. APPROACH

In this section, we clarify the basic algorithm, framework of our recommendation system and the recommending strategy we used.

A. Similarity Computation

We need to get the similarity between bug reports in both online and offline stages. It's also used as input of the portfolio based algorithm in the last step. So in this section, we clarify the computational formula to get it.

Each of the bug reports in the repository is turned into a word count vector and a topic vector. The words in bug reports are collected in a dictionary, and each of the word is assigned with a unique label. Formally, w_a and t_a are in the form of $w_a(w_{a,1}, w_{a,2}, \dots, w_{a,i}, \dots, w_{a,n})$ and $t_a(t_{a,1}, t_{a,2}, \dots, t_{a,i}, \dots, t_{a,n})$, where a refers to the bug report a, $w_{a,i}$ means the times that the word i appears in the bug report a, and $t_{a,i}$ means the possibility that the bug report a belongs to the ith topic.

With these two vectors, we calculate similarities between the new bug report and each bug report in the repository, which include *SimilarityW(a,u)* and *SimilarityT(a,u)*. They are the similarity between the word vector and topic vector of the bug report a,u respectively. The word vector similarity between bug report a and u is defined as

$$SimilarityW(a,u) = \frac{2 \times |I_a \cap I_u|}{|I_a| + |I_u|}, \quad (1)$$

where $|I_a \cap I_u|$ is the total count of words that appear in both the two bug reports, and $|I_a|$ and $|I_u|$ are the total number of words appearing in bug reports a and u, respectively. Both of the number of words that bug report a and u share and their corresponding counts have an influence on the similarity. When the co-appearing word count $|I_a \cap I_u|$ is small, the significance weight *SimilarityW(a,u)* will decrease the similarity estimation value between the bug reports a and u. Since the value *SimilarityW(a,u)* is between the interval of [0,1], the closer the value is to 1, the more similar bug reports a and u are. The similarity between the topic possibility vectors is defined as

$$SimilarityT(a,u) = \sqrt{\sum (t_{a,i} - t_{u,i})^2}, \quad (2)$$

where i is the label of the topic, $t_{a,i}$ and $t_{u,i}$ refer to the possibility that bug report a and u belongs to topic i,

respectively. We take the cosine distance between them as their distance. The interval of $SimilarityT(a,u)$ is $[0,+\infty]$. To normalize the topic similarity, we take its reciprocal and add 1 to the denominator avoiding the similarity to be infinite. Thus, the topic similarity is

$$SimilarityT(a,u) = \frac{1}{SimilarityT(a,u) + 1}. \quad (3)$$

We can see the range of similarity of topic $SimilarityT(a,u)$ is $[0,1]$, which is comparable with $SimilarityW(a,u)$.

B. Recommendation Framework

Here, we first define the problem of developer recommendations with similarity and cost awareness, and then clarify the recommendation framework.

DEFINITION 1 (PROBLEM STATEMENT). *Given a new bug report b , the goal of recommendation with accuracy and cost awareness is to build an optimal ranked list of developers based on both the developers' possibility of solving the problem and the time they need to solve it.*

The above problem statement raises two issues:

- How to get developers' possibilities of solving the bug report and produce a ranked list.
- How to combine the risk-based rank list with the time-based ranked list to produce final ranking to strike a balance between accuracy and cost, which means lessen the time without much influence to the accuracy.

There are often many developers as candidates for recommendations. Thus, how to efficiently manage developers for recommendation is also an open question. To that end, we propose a novel recommendation framework to solve these problems.

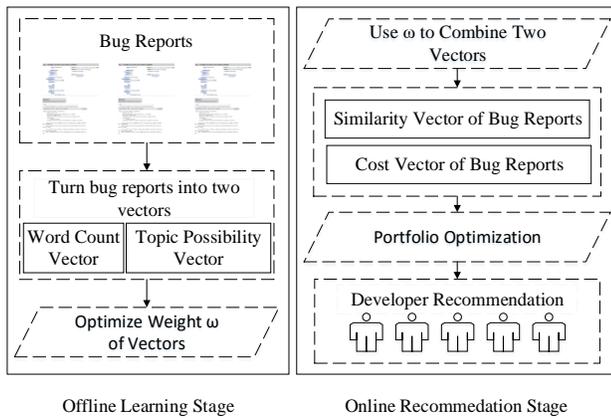


Figure 2. The recommendation framework

As we can see in Fig. 2, the whole recommendation framework contains two stages. The first stage is offline learning stage, where we learn the value of parameters that is needed in the second stage. And in online recommendation stage, we use modern portfolio theory [21] to combine the accuracy with the time cost to make recommendation about the best developer to solve the bug report. The specific method we apply will be clarified in the following specifications.

1) Offline Learning Stage

In this stage, we focus on determining the weight between the word count vector and the topic possibility vector.

These two vectors are different. Their contribution to the final prediction should differ too. So we need to determine their influences on the final prediction result.

For each bug report in testing set, we calculate their similarities and recommend the developer of the bug report which has the biggest similarity, and get the count of right recommendations. By varying the value of ω , we observe the change of the prediction similarity and take the value of ω when the similarity is the highest.

Algorithm 1 Automatic Detection of weight ω

Input: Bug reports training set $B1 = \{B1_i\}$; Bug reports testing set $B2 = \{B2_i\}$; Step size ε ; Developer Set $D = \{D_i\}$;

Output: The weight ω

1. Calculate $SimilarityW$ and $SimilarityT$ in $B1$ and $B2$
2. $maxcount = 0$;
3. for each $\Delta \in [0,1]$ with a step size ε , do
4. for each $B2_i \in B2$
5. $count = 0$;
6. get $B1_j$ with the biggest $Similarity(B1_j, B2_i)$
7. if $(D_i = D_j)$ then
8. $count ++$;
9. end if
10. end for
11. if $(count > maxcount)$ then
12. $maxcount = count$; $\omega = \Delta$;
13. end if
14. end for
15. return ω

2) Online Recommendation Stage

After we get the weight ω in the offline learning stage, we can carry on to the online recommendation stage. In this stage, ω is used to get the similarity vector between bug reports. With ω adjusting weights, the similarity between bug report a and u $Similarity(a,u)$ is

$$Similarity(a,u) = \omega SimilarityW(a,u) + (1 - \omega) SimilarityT(a,u). \quad (4)$$

$Similarity(a,u)$ ranges between 0 and 1. It implies ascending similarity with its value changing from 0 to 1. With $Similarity(a,u)$, we can get the accuracy vector:

$$Accuracy(u) = (accuracy_1, \dots, accuracy_i, \dots, accuracy_n) \quad (5)$$

$$accuracy_i = Similarity(a_i, u). \quad (6)$$

Each bug report has a time cost attribute which we use as the normalized cost of the bug report to form the cost vector, which is

$$Cost(u) = (cost_1, \dots, cost_i, \dots, cost_n). \quad (7)$$

where $cost_i$ is calculated as

$$cost_i = \frac{1/time_i}{1/\min(Cost(u))} \quad (8)$$

time_i is the time bug report i used to be fixed.

With these two vectors, the portfolio theory is applied to strike a balance between accuracy and cost to make the most appropriate recommendation by assigning a weight to each of the bug report. Ordering the bug reports by the weight ascendingly gets us the final recommendation result.

The detailed algorithm is shown in Algorithm 2.

Algorithm 2 Automatic Developer Recommendation

Input: Bug reports training set $B = \{B_i\}$; New bug report b ;

Output: The recommended developer D

1. Take bug reports from B with severity level higher than b as training set as B'
 2. Take each bug report in B' and get its cost and Accuracy to the new bug report as $Cost = \{Cost_i\}$ and $Accuracy = \{Accuracy_i\}$, respectively
 3. Apply modern portfolio theory and get a weight vector $W = \{w_1, w_2, \dots, w_n\}$ with w_i as the i^{th} bug report in B'
 4. Get the bug report with the largest w and take its fixer as D
 5. return D
-

C. Using Modern Portfolio Theory to Rank

When a new bug report comes up, it's useless to take bug reports with lower severity level as candidate recommender, for that the developer it recommends is highly impossible to fix the bug in time. So we take a strategy to filter out bug reports with lower severity level. This action is also beneficial for the instantaneity of recommendation. There are two types of ranking principles for recommendation.

- *Accuracy Principle*: We first rank bug reports in ascending order by their accuracy, and bug reports with the same accuracy will be further ranked by their cost.
- *Cost Principle*: We first rank bug reports in descending order by the cost, and bug report having the same cost will be further ranked by accuracy.

To make a finer recommendation structure, we need strike a balance between these two attributes, and thus modern portfolio theory [21] is used. This theory is first introduced in the field of portfolio investment. For example, there are n stocks; each stock has a future return with a risk. The theory aims at acquiring more future return with lower risk. In our problem, bug reports can be taken as the stocks, their accuracy can be regarded as the future return and their cost as risk. In specific, this theory assigns a weight to each of the bug reports which can be used to rank the bug reports. The bug report with the biggest weight should be the recommender.

Specially, a bug report portfolio can be represented by a collection of n bug reports with a corresponding weight assigned to each bug report b , i.e.

$$\gamma = \{(a_i, \omega_i)\}, \quad \text{s.t.} \quad \sum_i \omega_i = 1. \quad (9)$$

The weight ω in our problem indicates how much attention the recommendation system wants the user to pay on the bug report b_i . Therefore, the weights can be used to determine the ranks of bug reports; that is, bug reports should be ranked by the descending order of their weights. The future return of bug reports is $E[\gamma]$, which can be computed by

$$E[\gamma] = \sum_i \omega_i \cdot \Delta_i^{-1}. \quad (10)$$

where Δ_i is the rank of bug reports b_i in the accuracy based rank list. Also, the future risk of bug reports is defined as $R[\gamma]$, which can be computed as

$$R[\gamma] = \sum_i \left(\omega_i^2 \nabla_i^{-2} + 2 \sum_{j=i+1}^n \omega_i \omega_j \nabla_i^{-1} \nabla_j^{-1} J_{ij} \right). \quad (11)$$

where ∇_i is the rank^k of bug report b_i in the cost based ranked list, and J_{ij} is the risk correlation between Apps b_i and b_j . Here, we estimate J_{ij} as *Similarity*(b_i, b_j).

In our problem, the objective is to learn a set of bug report weights w for maximizing the accuracy and minimizing the cost, i.e.

$$\arg \max_w (E[\gamma] - b \cdot R[\gamma]). \quad (12)$$

where b is a specified risk preference parameter, which can be defined as the given severity level in our experiments. The simplest way is to set b to a default value of 1, which equalizes the importance of the accuracy and the cost. We leave method defining the value of b under different circumstances to future work. In the experiment, we vary the value of b to see the relationship between accuracy and cost.

IV. EXPERIMENTAL SETUP

In this section, we evaluate the developer recommendation system with awareness of accuracy and cost (DRAC) approach with a large-scale real-world dataset.

A. Data Collection

This section presents our analysis on data we acquire and discuss the rationality of their appearance. We collect 484,870 bug reports from the project 'Eclipse' which is publicly accessible on the open source platform-Bugzilla. Those bug reports have different status. The status of bugs includes: unconfirmed, new, assigned, resolved, verified, closed and reopened. They represent different phases of bug fixing process. Only fixed bugs appear valid in the experiment. So we filter out non-fixed bug reports and that leaves 265,280 of them.

We mainly use the description of the bug reports by turning them into word count vectors and topic possibility vectors. First, we preprocess description by stemming, removing stop words, numerals and words with length over 20 characters. After preprocessing, bug reports become a set of words. To get the word count vector, dictionary is formed, and we take the count of the word in each bug report as the value of that word. For the topic vector, we apply LDA to the words to extract the corpus' topics and then calculate bug reports' possibility belonging to each of the topics as the topic vector. In this pro-

cess, we adopt a widely used LDA implementation LDA4⁵. We followed guides in [22] and set the number of topics to 17. The format of the topic is shown in TABLE 1. Each topic has a set of words which correspond to its possibility of appearing in this topic. We take the square error of word frequency as the similarity between a bug report and a topic.

TABLE I. EXAMPLE WORD DISTRIBUTION OF TOPICS

Topic 1		Topic 14		Topic 17	
property	0.043	org	0.042	file	0.092
persist	0.027	java	0.020	project	0.066
connect	0.018	eclipse	0.014	build	0.032
test	0.015	report	0.013	plugin	0.027
map	0.014	birth	0.012	create	0.022
query	0.013	apache	0.010	jar	0.018
jpa	0.010	engine	0.009	package	0.018
value	0.010	service	0.009	path	0.016
service	0.009	jetties	0.009	folder	0.014
session	0.008	invoke	0.009	workspace	0.013

B. Baseline: Costriage

To our best knowledge, there is only one similar research [23]. It brought up the method *Costriage*. *Costriage* follows three steps. First, it constructs the developer profiles, which is a numeric vector with each element denotes the developer’s estimated cost L_C for fixing a certain bug type. To fill in the blanks in developer’s profile, collaborative filtering is used to predict the missing value. Thus, it gets the developers’ cost to solve every bug type. Second, it trains a multi-class classifier, when a new bug came, it estimates each developer’s score for the new bug and form the vector L_S by extracting its feature vector and applying the classifier. L_C and L_S are

$$L_C = (s_{c[1]}, s_{c[2]}, \dots, s_{c[n]}), L_S = (s_{s[1]}, s_{s[2]}, \dots, s_{s[n]}). \quad (13)$$

where $s_{c[i]}$ means the i^{th} developer’s estimated cost to fix the given bug, and $s_{s[i]}$ denotes the i^{th} developer’s success possibility for fixing the new bug.

Third, L_C and L_S are merged into one vector L_H , and then the first developer is assigned to the new bug report.

$$L_H = (s_{H[1]}, s_{H[2]}, \dots, s_{H[n]}) \quad (14)$$

$S_{H[i]}$ is obtained by

$$s_{H[i]} = \alpha \cdot \frac{s_{s[i]}}{\max(L_S)} + (1-\alpha) \cdot \frac{1/s_{c[i]}}{1/\min(L_C)}, \quad (15)$$

where $0 \leq \alpha \leq 1$.

V. EXPERIMENTAL RESULTS

A. Bug Report Severity Levels

To see the relationship between topics and severity levels, we do statistics about the distribution of severity level with different topics. Fig. 3(a) shows the percentage of bug reports with different topics. The bug reports belonging to topic 14 take up the majority of bug report repository. Topic 10, 11, 17 is the following topic with relatively large volume. Fig. 3 (b)-(d) shows the percentage of bug reports with different severities in topic 3, 11 and 14, respectively. Based on our

results, “Normal” bug reports is the main component of all the topics with a proportion fluctuating around 70%. It would be hard to see other severity levels’ status with “Normal” in the figure. So we leave out bug reports with severity level “Normal” and clarify other severity levels’ distribution among topics. In these figures, we can see that topics have their bias towards different severity level. Topic 14, which has the most bug reports, contains bug reports with severity level ‘Enhancement’ and ‘Major’ mainly. Half bug reports belonging to Topic 11 are ‘Major’ which is much higher than the other two topics. And Topic 3 is mainly composed of ‘Enhancement’ bug reports. Different topics have slight bias towards different severity levels. Based on the statistical results, we can see that there is a correlation between topic and severity level preference. Different topics have different severity level components.

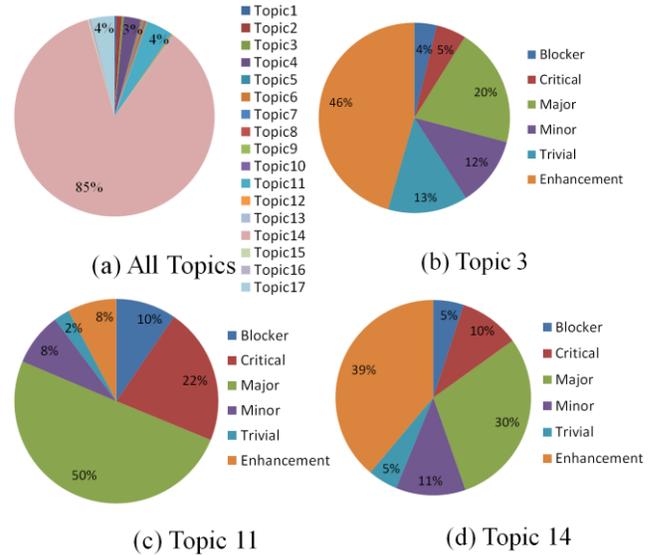


Figure 3. The percent of bug reports (a) and bug report topics at different severity levels (b)-(d)

B. Evaluation of Developer Recommendation

In the offline learning stage, to learn the weight between word vector and topic vector, we set the step length to 0.2. And accuracy spikes with 0.6. So we set the value of ω to 0.6. We order the bug reports in time series, and take the first 80% of them as training set with the remaining 20% as testing set. Specifically, we set up the evaluation as follows. First, we implement *DRAC* and *Costriage*. Then, we observe the trade-offs between bug assignment accuracy and bug fix time using *DRAC* compared to *Costriage*. We apply them with varying b to observe the trade-offs between accuracy and average bug fix time. Fig. 4 shows the comparison result. The x-axis represents the average time to fix one bug, and the y-axis represents bug assignment accuracy.

We can see from Fig. 4 that there is an obvious sign of trade-off. When the accuracy of the recommendation ascends, the average time to fix a bug increases with it, which is very understandable and confirms the idea we brought before. By altering the weight b , the developer can get different recommendation results with different purposes.

⁵ <https://github.com/hankcs/LDA4j/tree/master/src>

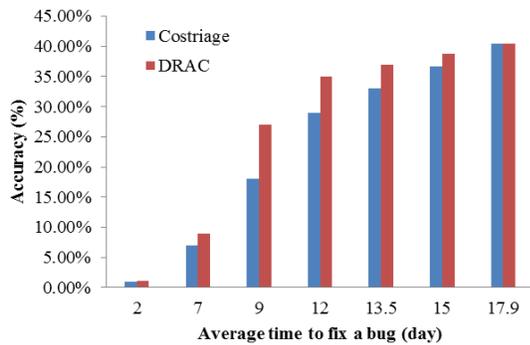


Figure 4. The trade-offs between accuracy and bug fix time

VI. CONCLUSION

In this paper, we developed a reviewer recommender system with awareness of accuracy and cost. Specifically, we learned a function to determine the weight of data from different sources. Moreover, to consider both reviewers' accuracy and cost, we introduced a flexible recommendation method based on modern portfolio theory. A key challenge in this research field is upholding the recommendation speed with big volume and high growth rate of the data. To address this problem, we propose *DRAC* to reduce the size of dataset and make recommendation considering the severity level and importance of bug reports. The experiments on a large-scale real-world dataset clearly clarified the effectiveness of the proposed recommendation framework.

Our recommendation framework is not restricted to developer recommendation. It can be applied to other field with similar objective. Take code fragment recommendation as example, it needs to consider both the applicability of the recommended code fragment (accuracy) and its security. These two factors correspond to the two objectives we have in this problem. So for problems with two optimization objectives like code recommendation, our recommendation algorithm is theoretically applicable to them.

REFERENCES

- [1] J. Anvik, L. Hiew, and G. C. Murphy, 2006. "Who should fix this bug?" In ICSE'06.
- [2] J. Zhang, X. Y. Wang, D. Hao, "A survey on bug-report analysis"[J]. Science China, 2015, 58(2):1-24.
- [3] Z. Xu, Y. Liu, Y. Xuan J, et al. Crowdsourcing based social media data analysis of urban emergency events[J]. Multimedia Tools & Applications, 2015:1-18.
- [4] Z. Xu, Y. Liu, N. Y. Yen, et al. Crowdsourcing based Description of Urban Emergency Events using Social Media Big Data[J]. IEEE Transactions on Cloud Computing, 2016:1-1.
- [5] T. T. Nguyen, A. T. Nguyen, T. N. Nguyen, "Topic-based, time-aware bug assignment"[J]. Acm Sigsoft Software Engineering Notes, 2014, 39(1):1-4.
- [6] T. Zhang, H. Jiang, Luo X, "A Literature Review of Research in Bug Resolution: Tasks, Challenges and Future Directions"[J]. Computer Journal, 2015.
- [7] S. Banerjee, J. Helmick, Z. Syed, "Eclipse vs. Mozilla: A Comparison of Two Large-Scale Open Source Problem Report Repositories"[C]// High Assurance Systems Engineering (HASE), 2015 IEEE 16th International Symposium on. IEEE, 2015:263-270.
- [8] C. L. P. Chen, C. Y. Zhang, "Data-intensive applications, challenges, techniques and technologies: A survey on Big Data[J]. Information Sciences, 2014, 275(11):314-347.
- [9] E. Savitz, Gartner, "10 Critical Tech Trends for the Next Five Years", October 2012.<<http://www.forbes.com/sites/eric savitz/2012/10/22/gartner-10-critical-tech-trends-for-the-next-five-years/>>.
- [10] J. Xuan, H. Jiang, Y. Hu, "Towards Effective Bug Triage with Software Data Reduction Techniques"[J]. IEEE Transactions on Knowledge & Data Engineering, 2015, 27(1):264-280.
- [11] P. Nagarnaik, A. Thomas, "Survey on recommendation system methods." In Electronics and Communication Systems (ICECS), 2015 2nd International Conference on, pp. 1496-1501. IEEE, 2015
- [12] Z. Bo, N. Lulian, G. A. Rajiv, "Cross-platform Analysis of Bugs and Bug-fixing in Open Source Projects Desktop vs. Android vs. iOS." In: Proceedings of the 19th International Conference on Evaluation and Assessment in Software Engineering, Nanjing, China, 2015.
- [13] J. Xuan, H. Jiang, Z. Ren, "Automatic bug triage using semi-supervised text classification." In: Proceedings of International Conference on Software Engineering & Knowledge Engineering, Redwood City, 2010. 209-214
- [14] G. Canfora, L. Cerulo. "Supporting change request assignment in open-source development." In: Proceedings of the ACM Symposium on Applied Computing, Dijon, 2006. 1767-1772
- [15] D. Matter, A. Kuhn, O. Nierstrasz, "Assigning bug reports using a vocabulary-based expertise model of developers." In: Proceedings of the International Working Conference on Mining Software Repositories, Vancouver, 2009. 131-140
- [16] G. Jeong, S. Kim, T. Zimmermann, "Improving bug triage with bug tossing graphs." In: Proceedings of the joint meeting of the European Software Engineering Conference and the ACM SIGSOFT International Symposium on Foundations of Software Engineering, Amsterdam, 2009. 111-120
- [17] A. Tamrawi, T. T. Nguyen, J. Al-Kofahi, "Fuzzy set-based automatic bug triaging." In: Proceedings of the International Conference on Software Engineering, Waikiki, 2011. 884-887
- [18] X. Xia, D. Lo, X. Wang, "Accurate developer recommendation for bug resolution." In: Proceedings of the Working Conference on Reverse Engineering, Koblenz, 2013. 72-81
- [19] H. Hu, H. Zhang, J. Xuan, "Effective bug triage based on historical bug-fix information." In: Proceedings of the IEEE International Symposium on Software Reliability Engineering, Naples, 2014. 122-132
- [20] M. Schulmerich, Y. M. Leporcher, C. H. Eu, "Modern Portfolio Theory and Its Problems"[M]// Applied Asset and Risk Management. Springer Berlin Heidelberg, 2015:101-173.
- [21] J. Wang and J. Zhu, "Portfolio theory of information retrieval." In *Proceedings of the 32Nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '09, pages 115{122, New York, NY, USA, 2009. ACM.
- [22] R. Arun, V. Suresh, C. E. V. Madhavan, M. N. N. Murthy, 2010. "On finding the natural number of topics with latent dirichlet allocation: Some observations." In PAKDD'10
- [23] J. W. Park, M. W. Lee, J. Kim, "CosTriage: A Cost-Aware Triage Algorithm for Bug Reporting Systems."[C]// Proceedings of the Twenty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2011, San Francisco, California, USA, August 7-11, 2011. 2011.

Managing forest transportation planning using ant colony optimization method

Pengpeng Lin

Math, Stat and Computer Science
University of Wisconsin - Stout
Menomonie, WI 54751, USA
linp@uwstout.edu

Ruxin Dai

Computer Science and Info Sys
University of Wisconsin - River Falls
River Falls, WI 54022, USA
ruxin.dai@uwrf.edu

Ran An

Plant & Soil Sciences
University of Kentucky
Lexington, KY 40506, USA
ran224@g.uky.edu

Abstract— Problems related to the transport of timber products from harvesting sites to conversion facilities have traditionally involved finding routes that minimize timber hauling and road construction costs. Increasing environmental concerns have introduced negative impacts, such as soil erosion and water quality, into forest transportation planning problems (FTPPs) making them larger and more complex. In this paper, we present an algorithm based on the ant colony optimization (ACO) framework, which has been shown to efficiently solve large-scale and complex multi-objective and constrained optimization problems. To test the performance of the algorithm, we used 10 different FTTP instances with various timber sale locations and destination nodes. Solutions were then compared with those obtained from comparable mixed-integer programming formulations solved by CPLEX. The experimental results showed that our algorithm was able to compete with CPLEX with similar solution quality for all test cases with significantly reduced computing times.

keywords- ACO; Forest-management; Transportation.

I. INTRODUCTION

The forest transportation planning problems (FTPPs) have focused on finding routes that minimize log hauling and road construction costs from harvesting sites to conversion facilities [1]. FTTPs that contain both variable (log hauling) and fixed costs (road construction) are a special case of the fixed charge transportation problem (FCTP), which is known as a NP-hard combinatorial optimization problem [2]. Mixed-integer programming (MIP) has been commonly used to optimally solve FCTP [3]; however, its application has been limited to small- and medium-scale problems because solution time grows exponentially with problem size [4]. Moreover, increasing environmental concerns related to the transport of timber products have introduced negative impacts such as erosion and water quality into FTTPs [5]. These environmental considerations and requirements introduce side constraints making FTTPs more complex than the traditional cost minimization problems.

Several heuristic approaches have been developed to solve large-scale problems in a reasonable time. Although these approximation techniques do not guarantee optimality, they can efficiently provide high-quality solutions for large and complex problems [6][7][8], [9]. In this paper, we present an algorithm based on the ant colony optimization (ACO) framework, which has been shown to efficiently solve large-scale and complex multi-objective and constrained optimization problems. We used the algorithm to solve FTTP with fixed and variable costs as well as side constraints. Sediment yields expected

to erode from road surfaces due to the heavy traffic of log trucks represented the negative environmental impact of timber transport and were considered as a side constraint. Our ACO algorithm used a two-stage process to find the least-cost set of routes from each timber sale location to selected mills resulting in a total sediment yield below a maximum allowable amount. During Stage I, the algorithm uses only sediment yield information on each road segment to quickly find feasible solutions. During Stage II, sediment and cost information per road segment are considered to select feasible solutions resulting in least-cost routes. After a solution is built, a 1-opt local refinement procedure was implemented to improve solution quality. The ACO algorithm was applied to 10 problem instances with different timber sale locations and destinations, which was created from a medium scale, grid-shaped hypothetical FTTP with 500 road segments (where traffic is allowed on both directions), 25 timber sale locations, and a single mill destination. Four cases with increasing levels of sediment constraints were considered, and an exhaustive parameter search process was conducted to select the best parameter combination for each case. Solutions were then compared with those obtained from comparable mixed-integer programming formulations solved by CPLEX [10].

The paper is organized as followings. In section II, we introduce a constrained forest transportation planning problem (CFTTP) that is considered in this study. In section III, we describe in detail the ACO developed for solving the CFTTP. In section IV, we depict the experiment setup and present experimental results. In section V, we draw a conclusion to this work and layout possible future works.

II. CONSTRAINED FOREST TRANSPORTATION PLANNING PROBLEM

We consider a CFTTP where the objective is to find the set of least-cost routes from timber sale locations to designated mill destinations while reducing an important environmental impact associated with timber transport [1]. We consider sediments expected to erode from road surfaces due to the traffic of heavy log-trucks as constraints. The CFTTP can be modeled using a network comprised of a set of nodes V and edges E representing road intersections and segments, respectively. Each edge in the network is associated with three attributes: fixed cost (FC), variable cost (VC), and sediment yield (Sed). The fixed cost (\$) for an existing road segment is a one-time road construction cost and/or maintenance cost that is activated when the road segment is used regardless of traffic level; the variable cost is the actual hauling cost (\$)

yield (ton/year) is detrimental to the forest ecosystem and is expected to erode from road segments that are in use (similar to the fixed cost).

In the CFTPP formulation, $S = \{s_1, \dots, s_m\}$ is the set of timber sale locations and $M = \{m_1, \dots, m_n\}$ is the set of mill destinations, where $S, M \subset V$. Each timber sale $s_i \in S$ has a predefined volume of timber to be delivered to a mill destination $m_j \in M$ across the road network: $(S \cup M, \Omega)$, where $\Omega \subset E$ is the set of routes connecting all timber sale locations to the destination mills. For each route $R_{s_i, m_j} \in \Omega$ connecting a given timber sale location s_i to a mill m_j to transport a predefined timber volume, VC_{s_i, m_j} is the total variable cost, FC_{s_i, m_j} is the total fixed cost, Sed_{s_i, m_j} is the total sediment amount, and B_{s_i, m_j} is a binary variable indicating whether the route is used in the final solution. The objective function of the CFTPP can then be defined as follows:

$$\text{Minimize : } \sum_{R_{s_i, m_j} \in \Omega} (VC_{s_i, m_j} + FC_{s_i, m_j}) \times B_{s_i, m_j} \quad (1)$$

while restricting the total sediment amount from the entire road network to be below a maximum allowable amount Sed_{max} .

$$\text{Constraint : } \left(\sum_{R_{s_i, m_j} \in \Omega} Sed_{s_i, m_j} \times B_{s_i, m_j} \right) \leq Sed_{max} \quad (2)$$

and the combined timber volumes arriving at mills must agree with the total timber volumes shipped out from the timber sales:

$$\sum_{i=1}^m Vol_{s_i} = \sum_{j=1}^n Vol_{m_j}, \quad (3)$$

where Vol_{s_i} and Vol_{m_j} are the timber volumes shipped from s_i and arriving to m_j , respectively. For examples and more detailed descriptions of the FCTP and CFTPP, please see [11][12].

III. ANT COLONY OPTIMIZATION

A. Ant Traveling Scheme

In our ACO algorithm, one ant is placed in each entry node (timber sale location). Ants move sequentially through adjacent nodes until the destination node (mill) is reached. After all ants have found the best routes connecting each timber sale to the selected destination node, one iteration is completed. That an ant selects what node to visit next is based on a random number and a transition probability calculated for each adjacent edge as follows:

$$P_{i,j}^t = \frac{\tau_{i,j}^\alpha \times \eta_{i,j}^\beta}{\sum_{i,k \in Nbr} \tau_{i,k}^\alpha \times \eta_{i,k}^\beta}, \quad (4)$$

where $P_{i,j}^t$ is the transition probability with which an ant select edge (i, j) during iteration t , Nbr the set of nodes adjacent to node i , α and β are parameters that control the relative importance of the pheromone trail intensity τ and the visibility values η . Trail intensity refers to the amount of pheromone in the edge and indicates how often the edge has been selected in previous iterations. Visibility is usually calculated as a value representing the a priori quality of selecting an edge. In our ACO algorithm, visibility values are calculated differently in stages I and II. During Stage I, ants are set to rapidly find feasible solutions without consideration of transportation costs

Thus visibility values on each edge are calculated as the reciprocal of the associated sediment yields. After a feasible solution is found, Stage II starts and visibility values on each edge are calculated based on the reciprocal of sediment amount, fixed cost, and variable cost. Formulas 5 and 6 show the resulting functions to calculate transition probability values on each edge during Stages I and II, respectively.

$$P_{i,j}^t = \frac{\tau_{i,j}^\alpha \times Sed_{i,j}^\beta}{\sum_{i,k \in Nbr} \tau_{i,k}^\alpha \times Sed_{i,k}^\beta} \quad (5)$$

$$P_{i,j}^t = \frac{(\tau_{i,j})^\alpha \times [\lambda \times NFCost_{i,j}^{-1} + (1-\lambda) \times Sed_{i,j}^{-1}]^\beta}{\sum_{i,k \in Nbr} (\tau_{i,k})^\alpha \times [\lambda \times NFCost_{i,k}^{-1} + (1-\lambda) \times (Sed_{i,k}^{-1})]^\beta} \quad (6)$$

in which the parameter λ is a weight used to balance the importance of cost and sediment values; $NFCost_{i,j}$ is the unit cost (summation of the fixed and variable costs per timber volume) for the edge (i, j) calculated as:

$$NFCost_{i,j} = \frac{FC_{i,j}}{\sum_{q \in Q} Vol_q} + VC_{i,j} \quad (7)$$

where Q is the set of routes that use the component $c_{i,j}$, $\sum_{q \in Q} Vol_q$ is the total timber volume transported through the component $c_{i,j}$.

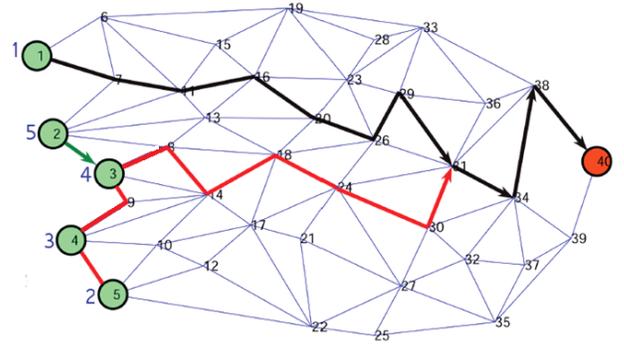


Fig. 1: Ant traveling scheme implemented into the ACO.

Ants move through adjacent nodes via edges, sequentially from each timber sale to the destination node. Starting at the first timber sale, an ant finds a route to the destination node. Then another ant starts from the next timber sale, and so on. The timber sale sequence is randomly determined every ACO iteration. During the route finding process, if an ant visits a node that is part of a previously found route, the ant stops the route finding process and the remaining route to the destination node attached to the current route. An example is shown in figure 1, where the sequence of timber sales (number next to green circles) is n_1, n_5, n_4, n_3, n_2 . Here, the selected route from the first timber sale to the destination ($n_1 \rightarrow n_7 \rightarrow \dots \rightarrow n_{38} \rightarrow n_{40}$) is shown as a black path. Then, while building a route from the second timber sale, a node part of a previous route if found (n_{31}), the ant stops moving through adjacent nodes, and the remainder of the route to the destination node is attached to the current route resulting in the following route: $n_5 \rightarrow n_3 \rightarrow \dots \rightarrow n_{31} \rightarrow n_{34} \rightarrow n_{38} \rightarrow n_{40}$. As the third and fourth timber sales in the sequence (n_4 and n_3 respectively) are already part of a previously found route, ants are not required

to find a new route. Thus the routes for the third and fourth timber sales are: $n_4 \rightarrow n_9 \rightarrow n_3 \rightarrow \dots \rightarrow n_{38} \rightarrow n_{40}$ and $n_3 \rightarrow n_8 \rightarrow \dots \rightarrow n_{38} \rightarrow n_{40}$. Lastly, the ant finding a route from the fifth timber sale to the destination makes only one move when a node part of a previous route is found. The resulting route is $n_2 \rightarrow n_3 \rightarrow \dots \rightarrow n_{38} \rightarrow n_{40}$. This ant travel searching mechanism is designed specifically for the FTTP with fixed and variable costs, where sharing edges by multiple timber sales often reduces the total fixed cost.

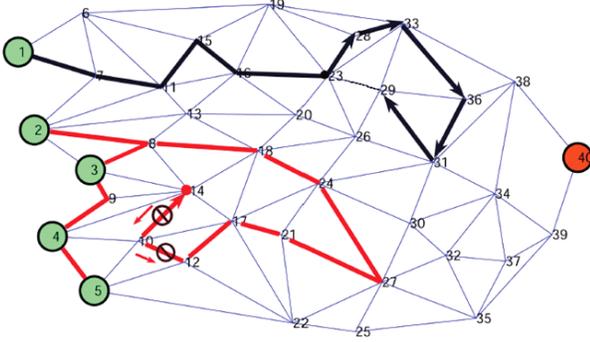


Fig. 2: Ant back-tracking process implemented into the ACO.

When searching for a route for a timber sale, the ant is set to ignore previously visited nodes along the current route to avoid forming circles. If, at a node, there is no available nodes to be selected because all adjacent nodes are already part of the current route, the node is marked as unavailable and the ant will move back through one edge to the previous node. This back-tracking process may have to continue until adjacent nodes become available. Figure 2 shows the case where an ant, after traveling to n_{29} , has only one node available (n_{26}) to visit next to avoid building a circle and another case where, after traveling to n_{14} , an ant has to back-track twice to n_{10} and then to n_{12} to have available nodes to visit next.

B. Local Search Refinement

In this study, a local search is implemented into our algorithm, which locally searches around the obtained solution every iteration for improvement in solution quality. Similar to the calculation of transition probability formulas 5 and 6, the local search seeks for a better solution based on sediment yield only during Stage I:

$$\min \sum_{\forall i,j \in E} Sed_{i,j} \quad (8)$$

and based on all three edge attributes during Stage II:

$$\min \sum_{\forall i,j \in E} \frac{FC_{i,j}}{\sum_{q \in Q} Vol_q} + VC_{i,j}. \quad (9)$$

After an iteration is completed, the local search looks at each node along the routes forming the solution and the adjacent nodes. For example, given a node n_i forming a solution route starting an origin node S , i.e., $(n_S \rightarrow \dots \rightarrow n_{i-1} \rightarrow n_i \rightarrow n_{i+1} \rightarrow \dots \rightarrow n_D)$ where n_D is the destination, the local search procedure looks at adjacent nodes of n_i other than n_{i+1} along the route and evaluates the edges to these nodes to determine if there is a shortcut that eliminates n_{i+1} from the route and provide a better solution in terms of

lower transportation cost. Figure 3 illustrates the local search process where a selected route (red path before refinement) is refined by the process resulting in the elimination of n_7 and n_{11} from the route (red path after refinement).

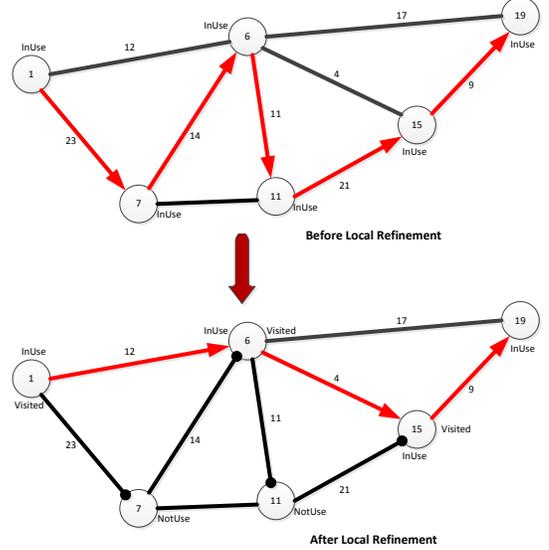


Fig. 3: Local refinement implemented into the ACO.

C. Pheromone Update

Pheromone evaporation is implemented in ACO algorithms to avoid a rapid convergence towards a suboptimal solution, allowing the exploration of other areas of the solution space. In our ACO algorithm, pheromone trail intensity is updated using the following formula:

$$\tau_{i,j}^{t+1} = \tau_{i,j}^t \times \rho + \Delta\tau_{i,j} \quad (10)$$

which considers two components. The first component $\tau_{i,j}^t$ is the current pheromone trail intensity on edge (i,j) at iteration t , which is multiplied by $0 < \rho < 1$, where $(1 - \rho)$ represents the pheromone evaporation between iterations t and $t+1$. The second component is the newly added pheromone amount to the edge (i,j) and is calculated differently in Stages I and II. During Stage I, with the purpose of obtaining feasible solutions, $\Delta\tau_{i,j}$ is calculated with sediment yield:

$$\Delta\tau_{i,j} = \begin{cases} \frac{Q}{Sed_{i,j}} & \text{Edge (i,j) is used} \\ 0 & \text{Otherwise} \end{cases}$$

and, during Stage II, $\Delta\tau_{i,j}$ is calculated based on all three edge attributes:

$$\Delta\tau_{i,j} = \begin{cases} \frac{Q}{\frac{FC_{i,j}}{\sum_{q \in Q} Vol_q} + VC_{i,j}} & \text{Edge (i,j) is used} \\ 0 & \text{Otherwise} \end{cases}$$

The Q in both stages is a constant with a value set to ensure that the amount of pheromone added to the edge (i,j) slightly increases the selection probability of the edge during the next iteration. In our experiments, Q was set to 0.00001.

D. Stopping Conditions

Three stopping criteria are implemented into our ACO algorithm to address solution quality stagnation and solving time efficiency. During the Stage I the algorithm tracks the number of iterations evaluated. If a user-defined maximum number of iterations (It_{sed}) is exceeded without finding a feasible solution, the algorithm stops and reports no feasible solution found. During Stage II, the algorithm tracks the number of consecutive infeasible iterations, and if it exceeds a user-defined maximum number (It_{infeas}), then the algorithm stops and reports the best feasible solution found. Each time a feasible solution is found, the algorithm resets the associated counter to zero. Also during Stage II the algorithm tracks the number of consecutive feasible solutions with inferior quality than the best solution found so far, and if it exceeds a user-defined maximum number (It_{feas}), the algorithm stops and the best feasible solution found is reported. In the experiments, It_{sed} , It_{infeas} , and It_{feas} were all set to 10,000.

IV. EXPERIMENT

A. Experiment Setup

The MIP and ACO algorithm implemented in C++ were tested using the Lipscomb High Performance Computing Cluster (HPC) supported and maintained by the University of Kentucky Center for Computational Science. The computing nodes of HPC (Dual Intel E5-2670 of 8 Cores at 2.6 GHz with 64 GB of 1600 MHz RAM) were used with Linux Red Hat OS. The ACO algorithm was applied to ten problem instances created using a 500-edge FTTP (Figure 4) presented in Contreras et al. [1]. This hypothetical problem allows traffic in both directions (thus 1000 edges); it considers 25 timber sale locations with a total volume of $36,500 m^3$ to be delivered to one mill destination in a single period. Variable cost, fixed cost, and sediment yield per edge ranged from $\$0.01/m^3$ to $\$10/m^3$, from $\$0.1$ to $\$23,000$ for road construction and maintenance, and from 0.4 to 200 tons, respectively. We also considered four cases with increasing level of sediment constraint. Case I was a fixed charge problem that considered minimizing cost without a sediment constraint; cases II and III were constrained fixed charge problems that considered cost-minimization subject to increasing levels of upper-bound sediment constraints making case III more difficult to solve than case II; case IV was a simple route finding problem where the objective is to minimize total amount of sediment without considering cost.

ACO parameters have shown to have a significant effect on solution quality. Thus we conducted an exhaustive parameter search on the four test cases of the original FTTP in [1] to find the best value for the four parameters ($\alpha, \beta, \rho, \lambda$) in the ACO algorithm. The range of values for all the parameters is set to $(0, 1]$ with a step 0.05 for α, β, ρ , 0.1 for λ , totaling 80,000 parameter combinations. Each parameter combination was applied ten times and the combination providing the best solution on average was selected as the best parameter combination. This exhaustive parameter search was conducted for all four cases and resulted in a different best parameter combination for each case (Table I). The parameter combinations were then used for the ACO to solve the four cases of the ten problem instances. The same set of parameter combinations were used for all problem instances. We want to point out that this is by no means the best approach for searching the optimal parameter combinations. Moreover, different problem instances might require different parameter combinations to maximize

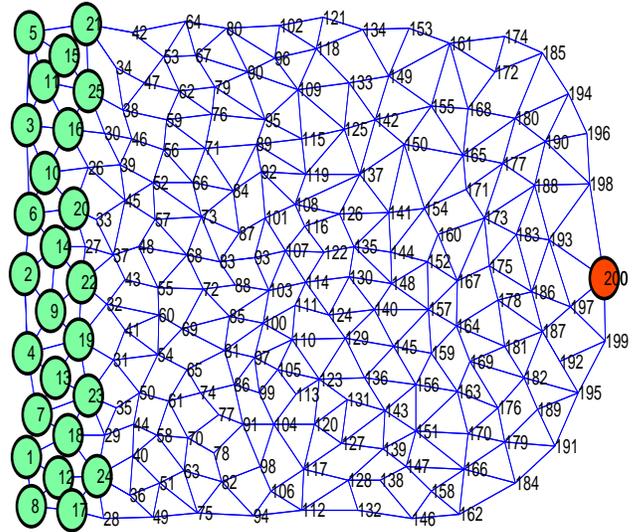


Fig. 4: An example of hypothetical FTTPs considered in this study.

ACO performance. However, the experimental results showed that ACO was able to obtain competitive objective function values for all test cases even with the same set of parameter combinations. How to optimize ACO parameter settings is an on-going open research topic and will be included in the future discussion.

TABLE I: Best parameter combination found.

Case	α	β	ρ	λ
I	0.5	0.4	0.55	1
II	0.5	0.9	0.6	0.7
III	0.5	0.7	0.6	0.7
IV	0.45	1	0.15	0

B. Experimental Results

The ACO algorithm was able to successfully solve all ten problem instances (Table II) with four cases for each of the ten instances. High quality solutions were obtained in many cases matching the optimal solutions from the MIP. For case I, the ACO algorithm was able to find optimal solutions for seven of the ten instances, and on average for all ten instances ACO solutions were 99.8% optimal. For case IV, the ACO algorithm found optimal solutions for all problem instances but number five, which solution quality was 99.6%. ACO solutions for the constrained FTTPs (cases II and III) also provided near-optimal solutions. For case II, ACO solution quality ranged from 97.7% to 99.9% with an average solution quality of 98.9%. As problem complexity increased in case III due to the more strict constraint, ACO solution quality averaged 97.8% with a range between 95.3% and 100%.

In terms of computing time, the ACO algorithm only required a fraction of time requested by MIP for all test cases (Table III and Table IV). On average, the best solution found

TABLE II: Objective function comparisons between MIP and ACO solutions for cases I, II, III and IV of the ten different problem instances.

Instance	Case	ACO (Objective Value - \$)	Sediment Constraint Value (tons)	MIP Objective Value - \$	Percent Different
1	I	866254	N/A	866254	0.00%
	II	887,719	2,159	878,749	1.02 %
	III	1,027,550	1,727	981,203	4.72 %
	IV	1294	N/A	1294	0.00%
2	I	1361070	N/A	1360965	0.01%
	II	1,416,090	2,490	1,415,960	0.01 %
	III	1,619,740	1,860	1,563,669	3.59 %
	IV	1230	N/A	1230	0.00%
3	I	1016500	N/A	1016500	0.00%
	II	1,055,330	2,254	1,048,768	0.63 %
	III	1,174,630	1,746	1,170,956	0.31 %
	IV	1236	N/A	1236	0.00%
4	I	897677	N/A	897677	0.00%
	II	914,972	2,449	910,152	0.53 %
	III	1,043,763	1,778	1,043,763	0.00 %
	IV	1106	N/A	1106	0.00%
5	I	1171130	N/A	1171130	0.00%
	II	1,203,500	2,445	1,181,284	1.88 %
	III	1,301,920	1,945	1,260,541	3.28 %
	IV	1389	N/A	1384	0.37%
6	I	1173387	N/A	1173387	0.00%
	II	1,212,620	2,354	1,208,610	0.33 %
	III	1,398,950	1,760	1,355,860	3.18 %
	IV	1163	N/A	1163	0.00%
7	I	1069100	N/A	1050671	1.75%
	II	1,089,140	2,672	1,066,148	2.16 %
	III	1,164,660	1,978	1,164,368	0.03 %
	IV	1283	N/A	1283	0.00%
8	I	1214400	N/A	1211218	0.26%
	II	1,241,760	2,660	1,229,392	1.01 %
	III	1,418,220	1,971	1,361,841	4.14 %
	IV	1343	N/A	1343	0.00%
9	I	1322930	N/A	1322930	0.00%
	II	1,410,850	2,262	1,378,432	2.35 %
	III	1,679,540	1,734	1,636,147	2.65 %
	IV	1205	N/A	1205	0.00%
10	I	1328930	N/A	1328930	0.00%
	II	1,403,150	2,342	1,394,355	0.63 %
	III	1,634,760	1,750	1,628,223	0.4 %
	IV	1132	N/A	1132	0.00%

by the ACO algorithm required about 25% (18 vs. 79 sec) and 1% (24 vs. 1678 sec) of the computing time required by the MIP solver to find the optimal solution for cases I and IV. For case II, the ACO was relatively similar between each instance, taking between 190 and 960 sec with an average of 544 sec. By contrast, computing time required by the MIP solver was about 18 times larger (9,949 sec). Due to the complexity of the problem, ACO and MIP solution times for case III were on average much larger and more variable than those for case II. ACO solution times varied from 360 to 29,000 sec with an average of 5,370 sec, which is only about 7.3% of the average time required by the MIP solver.

TABLE III: Computing times (Sec) for a single run required by the ACO algorithm for the four cases of each of the ten instances.

Instance	ACO (Sec)			
	Case I	Case II	Case III	Case IV
1	17	434	363	23
2	14	263	2,732	25
3	16	790	428	25
4	21	708	396	23
5	18	190	29,051	19
6	21	304	8,885	24
7	21	905	371	28
8	20	371	10,722	31
9	14	509	332	20
10	20	962	419	26
Average	18	544	5,370	24

TABLE IV: Computing times (sec) for a single run required by the MIP solver for the four cases of each of the ten instances.

Instance	MIP (Sec)			
	Case I	Case II	Case III	Case IV
1	41	3,254	31,722	2,790
2	134	62,314	90,973	500
3	65	3,532	21,893	2,147
4	95	4,732	36,385	1,275
5	64	3,110	149,585	54
6	69	4,344	97,582	2,922
7	76	1,333	31,540	2,933
8	68	8,458	55,516	1,786
9	84	1,182	152,629	1,157
10	98	7,225	59,597	1,212
Average	79	9,949	72,742	1,678

V. CONCLUSION

In this study, an ACO algorithm was developed to solve a CFTPP considering fixed and variable costs as well as a sediment constraint representing the negative environmental impact

of timber transport. The ACO was tested in ten problem instances with four different test cases. An exhaustive parameter search was conducted on the hypothetical network considered in this study to ensure the performance. Experimental results showed that the ACO was able to obtain near-optimal solutions and improved computing times significantly compared to the MIP.

The developed ACO algorithm has a great application potential to ensure the economic efficiency of timber transport operations, which is the largest cost component of timber harvesting. However, the ACO algorithm needs improvement to ensure solution quality and time efficiency for larger and more complex, real-world FTTP. In addition, configuring ACO parameters is a time consuming and tedious task. Future work should consider developing time efficient technique to configure parameter values without the need to conduct an exhaustive search. The current version of the algorithm coded in a sequential fashion, thus incorporating parallelization is likely to reduce solution time and allow addressing large-scale problems.

VI. INTRODUCTION

REFERENCES

- [1] M. A. Contreras, W. Chung, and G. Jones, "Applying ant colony optimization metaheuristic to solve forest transportation planning problems with side constraints," *Canadian journal of forest research*, vol. 38, no. 11, pp. 2896–2910, 2008.
- [2] D. I. Steinberg, "The fixed charge problem," *Naval Research Logistics Quarterly*, vol. 17, no. 2, pp. 217–235, 1970.
- [3] V. Adlakha and K. Kowalski, "A simple heuristic for solving small fixed-charge transportation problems," *Omega*, vol. 31, no. 3, pp. 205–211, 2003.
- [4] K. Kowalski, "On the structure of the fixed charge transportation problem," *International Journal of Mathematical Education in Science and Technology*, vol. 36, no. 8, pp. 879–888, 2005.
- [5] J. M. Grace III and B. D. Clinton, "Protecting soil and water in forest road management," 2007.
- [6] D. L. Martell, E. A. Gunn, and A. Weintraub, "Forest management challenges for operational researchers," *European journal of operational research*, vol. 104, no. 1, pp. 1–17, 1998.
- [7] P. Lin, J. Zhang, M. Contreras *et al.*, "Applying pareto ant colony optimization to solve bi-objective forest transportation planning problems," in *Information Reuse and Integration (IRI), 2014 IEEE 15th International Conference on*. IEEE, 2014, pp. 795–802.
- [8] P. Lin, J. Zhang, and M. A. Contreras, "Automatically configuring ACO using multilevel paramils to solve transportation planning problems with underlying weighted networks," *Swarm and Evolutionary Computation*, vol. 20, pp. 48–57, 2015. [Online]. Available: <http://www.sciencedirect.com/science/article/pii/S2210650214000789>
- [9] P. Lin, M. A. Contreras, R. Dai, and J. Zhang, "A multilevel ACO approach for solving forest transportation planning problems with environmental constraints," *Swarm and Evolutionary Computation*, vol. 28, pp. 78–87, 2016. [Online]. Available: <http://www.sciencedirect.com/science/article/pii/S2210650216000134>
- [10] I. I. CPLEX, "V12. 1: Users manual for cplex," *International Business Machines Corporation*, vol. 46, no. 53, p. 157, 2009.
- [11] N. Andalaft, P. Andalaft, M. Guignard, A. Magendzo, A. Wainer, and A. Weintraub, "A problem of forest harvesting and road building solved through model strengthening and lagrangean relaxation," *Operations Research*, vol. 51, no. 4, pp. 613–628, 2003.
- [12] M. L. Balinski, "Fixed-cost transportation problems," *Naval Research Logistics Quarterly*, vol. 8, no. 1, pp. 41–54, 1961.

Multi-Objective Biogeography-Based Method to Optimize Virtual Machine Consolidation

Kai Shi^{1,2}, HuiQun Yu(Corresponding Author)¹, Fei Luo¹, GuiSheng Fan¹

¹ Department of Computer Science and Engineering

East China University of Science and Technology, Shanghai 200237, China

² Shanghai Key Laboratory of Computer Software Evaluating and Testing, Shanghai 201112, China

BH4AWS@163.com, yhq@ecust.edu.cn, luof@ecust.edu.cn, gsfan@ecust.edu.cn

Abstract—Virtual machine consolidation (VMC) is an important issue in cloud computing, which can be used to reduce power consumption and achieve reasonable resource allocation. In this paper, an IMBBO algorithm is proposed to solve the multi-objective optimization problem of VMC through improving the classical Biogeography-Based Optimization (BBO). An improved Cosine migration model and an improved mutation model are presented to increase the efficiency of achieving the optimal solution. Meanwhile, three optimization objectives for server power consumption, load balancing and migration resource overhead are mainly addressed. Finally, several experiments are done to evaluate the performance of IMBBO by comparing with Gravitational Search Algorithm (GSA) based on the synthetic and real VM running data. The results show that the IMBBO optimizes VM consolidation with higher efficiency.

Keywords—multi-objective optimization; biogeography-based optimization; virtual machine consolidation; cloud computing

I. INTRODUCTION

CLOUD computing has been popular in the IT industry since 2008. The virtualization is the one of core technology in cloud computing. Generally, data centers use the deployed servers to provide different services and it will underutilize the server and increase the energy costs. A great number of virtual machines (VMs) which are randomly deployed in data centers in order to provide the service for end-users according to the required resources. However, this random deployment may bring about serious power consumption and resource contention. Meanwhile, the load unbalancing is also urgent to be solved. For example, data centers consumed about 1.5% of the total generated electricity in the U.S. in 2006 according to a report published in 2008. In fact, a great number of physical machines (PMs) are in idle state.

Recently, many researchers and institutes have focused on virtual machine consolidation problem, which can effectively reduce the energy consumption in one data center by consolidating VMs to PMs and shut down the idle ones. However, most methods just consider how to consolidate some VMs to some PMs, it will result in the unreasonable resource contention because VMC is a dynamic process. For the cloud providers, a good VMC scheme should maximize resource utilization and minimize power consumption and others.

Since the traditional VMC scheme just focus on how to minimize the consolidation, the similarity among VMs resource utilization isn't considered. More VMs may be migrated to the same PM that will extremely result in the unstable state of PMs and other VMs. In additional, VMs migration not only occupies the network bandwidth, but also the CPU utilization in source and destination. PMs may cause extra overhead such as more CPU cycle, the high memory operation and network transfer. Therefore, these factors should be considered in VMC in order to save power consumption.

In order to solve the above objective optimization problem, a novel multi-objective VMC algorithm named IMBBO (Improved BBO) is proposed based on improving the classical BBO algorithm. **The major contributions of this paper are as follows.**

1. The VMC problem is formulated as a multi-objective optimization problem, which includes the novel minimizing server power consumption, achieving better loading balance based on VMs' resource correlation, and reducing migration resource overhead considering in source and destination PMs.

2. A novel multi-objective optimization algorithm, IMBBO, is presented. The key factors of migration and mutation model in IMBBO are modified in order to preferably adapt to the VMC problem. Meanwhile, based on synthetic and real VMs running data, we compare the proposed IMBBO with Gravitational Search Algorithm (GSA), thus explaining the advantage of our method.

The rest of this paper is organized as follows. Section II discusses related works. Section III formulates the VMC as a multi-objective optimization problem. Section IV introduces the proposed algorithm, including its main process and key strategies. The simulation results are presented in Section V. And Section VI is the conclusion.

II. RELATED WORK

Recently, VMCP is one the well research area in cloud computing. Since, this problem is the NP-hard problem, many researchers from all world make their vastly effort to explore the approximate optimal solution. Generally, the VMCP can be

classified into two categories in terms of consolidation type, including static and dynamic ^[1].

The first type of research focuses on static single-objective virtual machine consolidation in tradition. Some research was based on off-line historical data and analyses the running feature in order to make a correct decision. E.g. the famous simulation software cloudSim^[2], improving energy efficiency ^[3], etc. These algorithms usually use threshold values to judge the running status of VMs or PMs. Furthermore, these researches just focus on single-objective consolidation optimization. However, in really, there is the correlation among all kinds of resources. Many extra conditions also should be considered in process. E.g. Fei Xu et al. and Raja Wasim Ahmad et al. wrote a survey about managing performance overhead of VMs and VMs migration in cloud ^{[1][4]}, which definitely summaries these features and correlation.

However, these static methods can't perfectly satisfy the rapid growth in the current data center environment. So the dynamic multi-objective VMC becomes a great attention ^{[1][4]}. Since many VMCP just solve a single objective, such as resource utilization, power consumption, etc. But, real VMC solutions often need to consider multi objectives and make a real time decision. E.g. Nguyen Trung Hieu et al. presented a VMCUP algorithm for improving the energy efficiency which dynamically predicts CPU utilization ^[6]. Chaima Ghribi et al. used B-matching algorithms to minimize the numbers of servers ^[5]. Chaima Ghribi et al. proposed the allocation and migration algorithms to minimize overall energy consumption ^[7]. Changming Zhao et al proposed a novel algorithm named Segmentation Iteration Correlation Combination (SICC), which integrates the methods of statistic regression modeling in order to reduce the difference of peak-mean value of VM resource utilization ^[8].

In recent years, other population research usually adopts heuristic algorithms to solve VMCP, such as heuristic bin packing, Biology-based optimization ^{[9][10]}, simulated annealing optimization, etc. In bin packing, the problem was formulated as a variant of vector bin packing problem, E.g. First Fit, Best Fit, Next Fit and Best Fit decreasing, etc. Furthermore, The GABA ^[11], the ant colony methods ^[12], SAPSO ^[13] and Simulated Annealing algorithm ^[14] are also represented. The ant methods used to pack VMs to the least number of PM necessary for the current workload. The SAPSO^[13] focuses on self-adaptive particle swarm. The GABA^[11] maps the VMs according to estimated future workload. Antonio Marotta et al. focus on a simulated annealing base algorithm which solves VMC by evaluating the attractiveness of the possible VM migrations, etc.

Since this paper focuses on the classical BBO algorithm, Biogeography-based optimization (BBO) is a new evolutionary algorithm firstly proposed in 2008 ^[15] and is an extension of biogeography theory to evolutionary algorithm ^[16], which is based on the mathematical model of biological species distribution and migration^[17]. The BBO has demonstrated good performance on various unconstrained and constrained benchmark functions ^{[18][19]}. Further, it has been applied to real world optimization problems, including sensor selection ^[15], power system optimization ^[20], etc.

Recently, some extension researches also have been presented. Typically, Haiping Ma ^[21] improved the classical BBO algorithm and analyzes the equilibrium of migration models. Haiping Ma and Dan Simon ^[22] discussed migration models using markov theory and blended BBO for constrained optimization. In real, BBO maps these factors as suitability index variable (SIV) and habitat suitability index (HSI) to mathematical solution space in order to find the optimal solution of a certain problem.

III. PROBLEM FORMULATION

The aim goal of this paper is to study the VMCP through multi-objective optimization which is based on IMBBO. To formulate this problem we will deal with it in the mathematical forms:

- N : The number of virtual machines in data centers (VMs)
 - M : The number of physical machines in data centers (PMs)
 - v_i : $i \in \{1, 2, 3, \dots, N\}$, The virtual machine of label i
 - p_j : $j \in \{1, 2, 3, \dots, M\}$, The physical machine of label j
 - ass_{ij} : The binary value represents whether virtual machine v_i is assigned to physical machine p_j
 - V : The set of virtual machines, namely $v_1, v_2, v_3, \dots, v_N$
 - P : The set of PMs, namely $p_1, p_2, p_3, \dots, p_M$
 - e_j^{busy} : The energy consumption of p_j , when $u_j = 100\%$
 - e_j^{idle} : The energy consumption of p_j , when $u_j = 0\%$ (just running the OS system)
 - v_{cpu}^i : The CPU demand of v_i
 - v_{mem}^i : The memory demand of v_i
 - v_{net}^i : The network demand of v_i (bandwidth)
 - p_{cpu}^j : The CPU capacity of p_j
 - p_{mem}^j : The memory capacity of p_j
 - p_{net}^j : The network capacity of p_j (bandwidth)
 - S_{kl}^{cpu} : The coefficient of similarity by CPU utilization between v_k and v_l
 - S_{kl}^{mem} : The coefficient of similarity by memory utilization between v_k and v_l
 - S_{kl}^{net} : The coefficient of similarity by network utilization between v_k and v_l
 - $cpu(pm_j)$: The current CPU utilization of pm_j
 - $mem(vm_i)$: The memory size of vm_i
 - $net(pm_j)$: The current NET bandwidth of pm_j
 - ∂_{ex} : The extra CPU utilization coefficient.
 - T_{stop} : The stop time of virtual machine migration.
- VP is a matrix which presents the allocation of VMs to PMs. Each element contains two types of value. If $ass_{ij} = 1$, v_i is assigned to p_j . Otherwise, v_i isn't assigned to p_j , where $1 \leq i \leq N, 1 \leq j \leq M$.

$$\text{Min} \sum_{j=1}^M \text{PowerConsumption}_j = \sum_{j=1}^M \left[y_j \times \left(e_j^{\text{idle}} + (e_j^{\text{busy}} - e_j^{\text{idle}}) \times \frac{\sum_{i=1}^N \text{ass}_{ij} \cdot (v_{cpu}^i + \delta_1 \cdot v_{mem}^i + \delta_2 \cdot v_{net}^i)}{P_{cpu}^j} \right) \right] \quad (1)$$

$$\text{MinLoadingBalance} = \sum_{j=1}^M \left[y_j \times \left(\sigma_{cpu} \cdot \frac{\sum_{k \neq j} S_{kl}^{cpu}}{C_{vmcount \in P_j}^2} + \sigma_{mem} \cdot \frac{\sum_{k \neq j} S_{kl}^{mem}}{C_{vmcount \in P_j}^2} + \sigma_{net} \cdot \frac{\sum_{k \neq j} S_{kl}^{net}}{C_{vmcount \in P_j}^2} \right) \right] \quad (2)$$

$$\text{min MigrationOverhead} = \sum_{j=1}^M \left[y_j \times \left(\sum_{i=1}^N \left(\text{cpu}(pm_i) \cdot \partial_{ex} + T_{stop} + \frac{\text{mem}(vm_i)}{\text{net}(pm_j)} + \text{cpu}(pm_i) \cdot \partial_{ex} + \frac{\text{mem}(vm_i)}{\text{net}(pm_j)} \right) \right) \right] \quad (3)$$

$$\text{s.t.} \begin{cases} \forall i, \sum_{j=1}^M v_{cpu}^i \cdot \text{ass}_{ij} \leq \sum_{i=1}^N P_{cpu}^j \cdot y_j; \sum_{i=1}^N v_{mem}^i \cdot \text{ass}_{ij} \leq \sum_{i=1}^N P_{mem}^j \cdot y_j; \sum_{i=1}^N v_{net}^i \cdot \text{ass}_{ij} \leq \sum_{i=1}^N P_{net}^j \cdot y_j \\ \sigma_{cpu} + \sigma_{mem} + \sigma_{net} = 1; \text{ass}_{ij} \in \{1, 0\}; i \in \{1, 2, 3, \dots, N\}, j \in \{1, 2, 3, \dots, M\} \end{cases} \quad (4)$$

$$\sigma_{cpu} + \sigma_{mem} + \sigma_{net} = 1; \text{ass}_{ij} \in \{1, 0\}; i \in \{1, 2, 3, \dots, N\}, j \in \{1, 2, 3, \dots, M\} \quad (5)$$

As above, **Eq.(1)-Eq.(3)** illustrate the objectives of VMCP and **Eq.(4)-Eq.(5)** are the total constraint condition for these objectives. These demonstrate that all required resources aren't greater than the capacity of one PM.

The first objective **Eq.(1)** is the minimization of server power consumption. The popular power consumption model has been introduced in ^{[9][10]} which shows the power consumption is linearly proportional on CPU utilization. However, the classic equation just pays attention to the CPU utilization. The operation of memory and network relating with the CPU cycle doesn't be considered. So more detail information of the new power consumption function is defined which is divided into static and dynamic parts in this paper. When the physical machine doesn't run any tasks (just OS system), the parameter e_{idle}^j is defined in idle status. In addition, the configuration parameters δ_1 and δ_2 present the weight coefficient that the memory and network operation take the proportion of CPU cycle. Finally, this scenario illustrates that PMs can be turn off, when these consume no extra energy.

The second objective **Eq. (2)** is the load balancing based on minimization correlation of resource utilization among PMs in the data center. The phenomenon shows that the load balancing is the similarity average resource utilization among PMs. Since the resources competition will result unstable resource condition among PMs and VMs, the reasonable solution is that these VMs with minimum similarity of resource utilization should be consolidated into same PM. In addition, the configuration parameters, σ_{cpu} , σ_{mem} and σ_{net} , are the weight coefficient, which is satisfied with constraint condition $\sigma_{cpu} + \sigma_{mem} + \sigma_{net} = 1$. Next, the $C_{vmcount \in P_j}^2$ is the permutation and combination in order to get average values of similarity.

The third objective **Eq. (3)** is the minimization of migration resource overhead in data center. When one VM migrates from pm_j to pm_k , the resource overhead may contain the CPU consumption and network bandwidth consumption which is both in source and destination PMs. Besides, the stop time of VMs and migration times, for example, $\text{mem}(vm_i)/\text{net}(pm_j)$ also should be considered in process. The parameter ∂_{ex} is the proportionality coefficient that the VM migration process may extra occupy the CPU cycle.

IV. IMPROVED BIOGEOGRAPHY-BASED OPTIMIZATION

In this section, the brief classical BBO algorithm will be introduced which includes the theory and the important features. Then, the improvement Cosine migration rate model and mutation rate model will be discussed in detail. Finally, the process of IMBBO will be described by the pseudo-codes.

A. Biogeography-Based Optimization

The classical Biogeography-based optimization algorithm is introduced by Simon in ^[15], which is based on the mathematics of biogeography theory and is a population global optimization approach. This algorithm can guarantee convergence to the optimal solution, if it is given enough generations (iterations). Biogeography studies the geographical distribution of species. Among them, the most important feature is migration and extinction (mutation).

B. The main improvement in IMBBO

1) The Improved Cosine Migration Rate Model

In this paper, the improved migration model is used to represent the migration feature. The linear function in BBO is basic migration model in BBO, which can be used to share SIVs between habitats. However, the linear model is the theoretical model with ideal condition. When there are more or less species in one habitat, the change rate of immigration and emigration will trend to the steady state. Otherwise, the change rate is fierce in the real virtualization environment. So this situation can be analogized in VMCP.

Meanwhile, many VMs are placed in the same PM may greatly increase the probability of resource contention. Finally, some VMs should be migrated to other PMs in order to maintain the running performance. From this, the improved Cosine migration rate model can be illustrated in **Eq. (6)**

$$\begin{aligned} \lambda_k &= \frac{I}{2} \left(\cos\left(\frac{k}{n} \cdot \pi + \beta\right) + 1 \right) + \varepsilon \\ \mu_k &= \frac{E}{2} \left(-\cos\left(\frac{k}{n} \cdot \pi + \beta\right) + 1 \right) + \omega \end{aligned} \quad (6)$$

Where λ_k and μ_k are presented by the immigration and emigration rates of the number of species k. The I is the maximal immigration rate when the number of species is zero. The E is the maximal emigration rate, when the number of species is to the maximum. So we set I = E in order to decrease experiment complexity in this paper. In addition, the parameter

β is the negative trigonometric offset angle (typically between $-\pi/2$ and 0). It denotes the degree of temporary positive immigration rate feedback in classical BBO. Next, the parameters ε and ω respectively denote the balance values. The migration rate cure for a single habitat is illustrated in **Fig.1(a)**. Further, the character of curve is more similar to the VM consideration features.

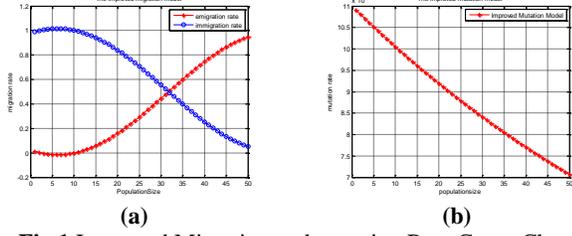


Fig.1 Improved Migration and mutation Rate Curve Chart

2) The Improved Mutation Model

The mutation model is another important feature in the classical BBO, which realizes mutation through a uniformly random function to probabilistically replace SIVs by randomly generating new SIVs in a solution. This situation can much better avoid search prematurity. However, the random function doesn't adapt to the VM running environment in real.

The improved descending function in this section, which is compared to the random function in BBO, is a more excellent strategy. With the increasing numbers of iteration, more excellent elitisms will be contained in next generation. The system will more tend to be a stable state. That is, the probability of mutation should be descended in iterations. The **Eq. (7)** represents the improved mutation model.

$$\mu_i = \tau \cdot \exp(-\sigma \cdot i) + \eta \quad (7)$$

Where the parameter τ is the slope value, it is defined by 0.01 in experiment. In addition, the parameter σ will be used to decide the degree of mutation rate in a time period. Next, the parameter η is the rectify coefficient, it is defined by 0.001 in experiment. The **Fig.1(b)** illustrates the improved mutation rate cure in a single habitat.

Pseudo-codes of IMBBO Algorithm	
Input:	the habitat matrix , migration, mutation model, objectives
Output:	the optimization solution and fitness
1:	Initialize the parameter and habitats(candidate solutions)
2:	While the stop condition is not satisfied
3:	Calculate HIS (fitness) for each habitat
4:	Update migration model and Calculate the rate value
5:	for each habitat(solutions) do
6:	for $i=1$ to maximum number of SIVs do Migration
7:	if $\text{rand} < \lambda_i$ (improved) then Selection
8:	for $j=1$ to maximum number of SIVs do
9:	if $\text{rand} < \mu_j$ (improved) then Emigration
10:	Sharing the SIV information from x_i to x_j
11:	end if
12:	end for
13:	end if
14:	end for
15:	end for
16:	for $i=1$ to all habitats do Mutation
17:	for $j=1$ to maximum numbers of SIVs do
18:	if $\text{rand} < \text{mutation probability}(\text{improved})$ then
19:	performing the mutation process
20:	end if
21:	end for
22:	end for
23:	process elitism and check the data legalization
24:	end while

Fig.2 the pseudo-codes of IMBBO

C. The analysis of IMBBO

The main process of IMBBO is similar as BBO. The pseudo-codes are as in **Fig.2**. Further, in order to preferably satisfy the VMCP and performance, the improved migration model, including immigration and emigration rate calculation, is modified in line 7 and 9. In addition, the improved mutation model is also modified in line 18.

The time complexity of IMBBO is as:

$$\text{iteration} * [O(1) + 2 * O(m) + 2 * O(mn)] \approx K * O(mn)$$

The m presents the habitat size and the n presets a number of independent variables called Suitability Index Variables, which simulate the habitat number and features in habitat. So it is proportional to $m*n$. In addition, the space complexity is $m*n$, which presents the matrix size in IMBBO.

V. EVALUATION

In this section, experiment environment will be described and the results will be evaluated by using two different instance types, including synthetic instance and real VM running instance. Then, many experiments comparing with GSA algorithm are carried out to verify the performance of our algorithm from different aspects. Finally, we analyze and discuss the experimental results in detail.

A. Experimental and Method

1) Experimental Environment

In the experiment, two types of experiment data are used to verify the performance of IMBBO algorithm which is compared with GSA. Our experiment environment is based on the real VM running environment for ECUST Virtual Cloud Laboratory System. Meanwhile, IMBBO algorithm will be deployed on Monitor Server. The Virtual Cloud Laboratory System was developed on OpenStack. This system will provide basic experiment courses for students. The cluster is composed of a numbers of Dell PowerEdge R730 which is viewed as the computing node. Each server consists 2 physical CPUs (IntelE5-2650 v3 2.3GHz, 25M) and 256GB main memory.

TABLE I
PARAMETER VALUES FOR IMBBO AND GSA ALGORITHM

Parameter	Value	Parameter	Value
Habitats Size:	50	Improved mutate model τ :	0.01
SIV numbers:	200	Improved mutate model σ :	0.01
Elite habitats:	2	Improved mutate model η :	0.001
Maximum immigration rate:	1	Iterations numbers:	1000
Maximum emigration rate:	1	Distance bound norm:	2
The configuration coefficient β	$\pm \pi/8$	G0(GSA):	100
The configuration coefficient $\varepsilon \ \omega$	-0.02	Alfa(GSA):	20
Maximum constant mutate rate:	0.04		

2) Experimental Method

The different configuration coefficient will be used to verify this IMBBO algorithm. Some parameter values of IMBBO algorithm and GSA are defined in **Table I**. In order to getting average and exact results, each set is optimized by running independently 20 times and the average value is reported for each result. The Gravitational Search Algorithm (GSA) will view as the comparing algorithm for verifying the performance of IMBBO algorithm. Meanwhile, in order to keep the performance, two algorithms use the same set of initial data.

In the synthetic data: Servers are based on homogeneous framework. e_j^{idle} and e_j^{busy} are set to 162W and 215W. The

synthetic data is configured according to the reference [9]. The items in dataset independently follow the normal distribution, which has been adopted in previous researches, including CPU demands generated with $N(0.15,0.05)$, Memory demands generated with $N(0.10,0.08)$ and Network demands generated with $N(0.03,0.01)$.

TABLE II
AMAZON EC2 INSTANCE

Pattern	Instance Specs			Pattern	Instance Specs		
	Instance type	vCPU	Memory		Instance type	vCPU	Memory
General	T2.micro	1	1	General	M4.xlarge	4	16
	T2.small	1	2		M4.2xlarge	8	32
	T2.medium	2	4	Compute Optimized	C3.large	2	3.75
	M3.medium	1	3.75		C3.xlarge	4	7.5
	M3.large	2	7.5	Memory optimized	C3.2xlarge	8	15
	M3.xlarge	4	15		C3.4xlarge	16	30
	M3.2xlarge	8	30	R3.large	2	12.25	
	M4.largge	2	8	R3.xlarge	4	30.5	

(<http://aws.amazon.com/cn/ec2/instance-types/>)

In the real-world data: Sixteen types of instance (Table II.) referring to Amazon EC2 are also used to simulate in Virtual Cloud Laboratory System, which are divided into three categories including general, computing optimized and memory optimized. Since the reference just contains CPU and memory except network, the network parameter will be set through simulation or get by using the data collection tools which are developed by shell (Linux) or C# (Windows).

B. Experimental Results

1) Experimental Results of Synthetic Instances

In the synthetic data, there are 500 physical machines in one data center. The initial habitat is set with 500. The number of virtual machines is 200. Next, the initial power consumption is 500*162W and the detail result information is presented in Table III of synthetic data.

TABLE III
EXPERIMENTAL RESULT OF SYNTHETIC AND REAL VM RUNNING DATA

AI	Synthetic data				Real VM running data			
	AS	CC	SE	MRO	AS	CC	SE	MRO
BBO	99	18037.83	-0.3208	123511.72	30	5827.59	-0.0718	82015.62
IMBBO(1)	49	9937.83	-0.4351	14004.37	19	4045.58	-0.0891	3559.00
IMBBO(2)	92	16903.83	-0.3090	65954.81	31	5989.58	-0.0740	52782.19
GSA	96	10909.83	-0.2883	42013.09	33	4531.59	-0.0711	7118.01

(**AI:** algorithms, **AS:** active servers, **CC:** cost consumption, **SE:** similarity evaluation, **MRO:** migration resource overhead)

The improved migration rate model and mutation rate model are adopted in IMBBO(1). Meanwhile, the improved migration rate model and constant probability value of mutation model are adopted in IMBBO(2). Finally, the IMBBO(1) algorithm only needs 49 active servers to support these VMs running.

The IMBBO(1) can reduce the power consumption from 500*162 to 9937.83. That is, it saves 87.71% of power.

The similarity evaluation value of IMBBO(1) is -0.4351, which is greatly smaller than others. This situation demonstrates that the algorithm can gain better performance and achieve the smallest similarity among virtual machines in order to reduce resource competition.

The migration resource overhead value of IMBBO(1) reach 14004.37 through 1000 iterations. The CPU extra utilization and transfer times in source and destination PMs are considered in this process. Finally, the IMBBO(1) algorithm can reduce the resource overhead in migration process compared others.

Fig.3(a)-(c) show the comparisons of the three algorithms on synthetic data. The IMBBO algorithm rapidly finds the best solutions. The blue cure presents the BBO. The green cure is the IMBBO(2). The sky blue cure is the GSA. However, the red cure is the IMBBO(1).

In conclusion, the IMBBO algorithm shows the better performance both in reducing power consumption, achieving good load balancing (minimizing similarity values among VMs) and decreasing the migration resource overhead. The reason is that IMBBO algorithm considers the special migration model and dynamic consolidation judgment strategy. At the same time, the mutation rate trends to much more stabilization with the iteration. Besides, the different configuration parameters may display the different performance in the experiment. These parameters need to be adjusted according to the real VM running situation. Finally, the GSA algorithm just focuses on finding the massive particles. The correlation of iteration doesn't been contained through the iteration.

2) Experimental Results of Real VM running Instances

In the real VM running environment, we set 80 VMs which simulate 80 students to attend class in Virtual Cloud Laboratory System. The initial configuration needs 150 servers. The experiment results are described in Table III of real VM running data.

For experiment, The IMBBO(1) algorithm just needs 19 active servers to support these VMs running. The IMBBO(1) can reduce the power consumption from 150*162W to 4045.58. That is, it saves 83.36% of power. In addition, the similarity evaluation value of IMBBO(1) is -0.0891, which is greatly smaller than others in reducing resource utilization competition. Meanwhile, the migration resource overhead value of IMBBO(1) reach 3559.00 through 1000 iterations.

Fig.3(d)-(f) show the comparisons of the three algorithms on real VM running environment. These curve styles are the same as the synthetic instance. After the 1000 iterations, all of these curves approximately trend to line. Since a great number of VMs run in real data center environment, the appropriate running time and convergence time are the much more important parameter index.

Since these experiments are arranged based on homogeneous server architecture, some extra situations and different configuration parameters haven't been detailedly considered in process. It may exist others factors to influence the process of optimization. Besides, the emergency situation in some VMs also is the reason to affect optimization process.

Since the EA algorithms may lead to premature convergence (local optimization), including the BBO, the different strategies also should be adapted to avoid it. The appropriate parameters adjustment and the different situation analysis should be pre-done for the different objective functions.

VI. CONCLUSION

In this paper, the VMCP is formulated as a multi-objective optimization problem, which contains three configurable objectives, that is, server power consumption minimization, load balancing minimization based on resource utilization similarity and reducing migration resource overhead. A novel multi-objective optimization algorithm named IMBBO is proposed to solve VMCP based on the classical BBO. The

migration and mutation model was improved, which can better meet the actual situation of VMCP. The experiment results show that the IMBBO can improve the performance of optimal solutions and convergence characteristic by comparing with GSA by using the synthetic and real VMs running data.

In future, we will focus on parallelization of IMBBO and research re-configuration migration and mutation model in real problem. The appropriate adjustment also should be used to solve the different objective function in order to avoid shortcoming of premature convergence.

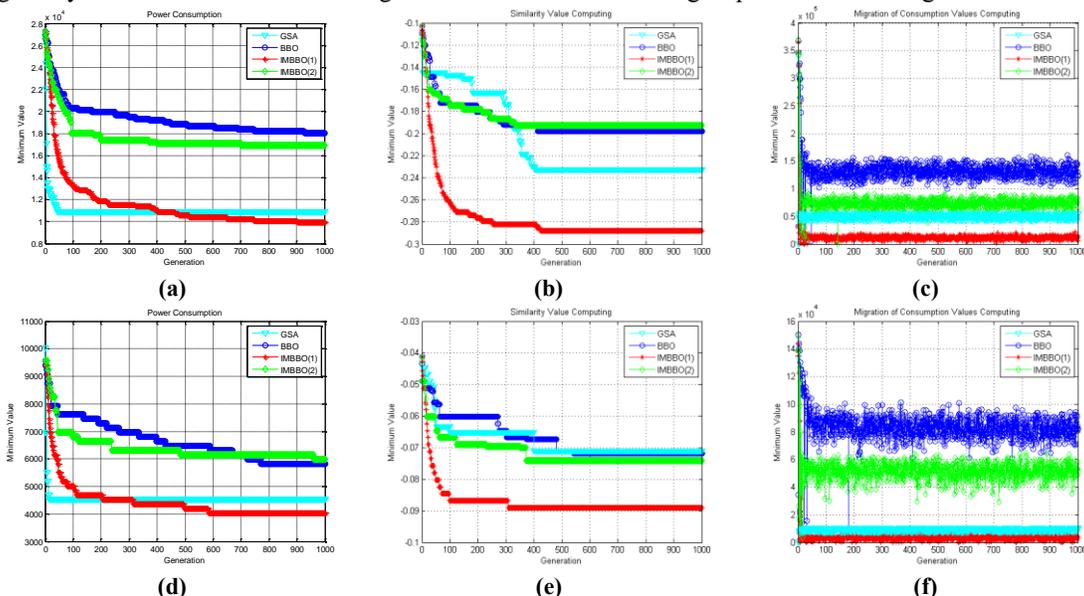


Fig.3 The experiment results of different comparing algorithms in synthetic and real VM running data

ACKNOWLEDGMENT

This research was partially supported by the Nature Science Fund of China under grants No. 61173048, 61300041, 61472139, Specialized Research Fund for Doctoral Program of Higher Education under grant No. 20130074110015.

REFERENCES

- [1] Xu, F., et al. "Managing Performance Overhead of Virtual Machines in Cloud Computing: A Survey, State of the Art, and Future Directions." *Proceedings of the IEEE* 102.1(2014):11-31.
- [2] Anton B. "Energy-Efficient Management of Virtual Machines in Data Centers for Cloud Computing" Ph.D. dissertation, Dept. Computing and Information Systems. Philosophy., Melbourne Univ., Melbourne, Australia, 2013.
- [3] Dabbagh, M., et al. "Energy-efficient cloud resource management," *Proc. IEEE INFOCOM Workshop on Mobile Cloud Computing (INFOCOM 2014)*, April 2014, pp. 386-391.
- [4] Ahmad, R. W., et al. "Virtual machine migration in cloud data centers: a review, taxonomy, and open research issues." *Journal of Supercomputing* 71(2015):1-43.
- [5] Hadji, M., et al. "A virtual machine repacking in clouds: faster live migration algorithms," *Proc. ACM SIGCOMM workshop on Distributed cloud computing (SIGCOMM 2014)*, Aug. 2014, pp. 37-38.
- [6] Hieu, N. T., et al. "Virtual Machine Consolidation with Usage Prediction for Energy-Efficient Cloud Data Centers." *Cloud Computing (CLOUD)*, 2015 IEEE 8th International Conference on. IEEE, 2015.
- [7] Ghribi, C., et al. "Energy efficient vm scheduling for cloud data centers: Exact allocation and migration algorithms." *Cluster, Cloud and Grid Computing (CCGrid)*, 2013 13th IEEE/ACM International Symposium on. IEEE, 2013.
- [8] Zhao, C., et al. "A Virtual Machine Dynamic Consolidation Algorithm Based Dynamic Complementation and FFD Algorithm." *Communication Systems and Network Technologies (CSNT)*, 2015 Fifth International Conference on. IEEE, 2015.
- [9] Zheng, Q., et al. "Virtual machine consolidated placement based on multi-objective biogeography-based optimization." *Future Generation Computer Systems* 54.C(2015):95-122.
- [10] Zheng, Q., et al. "Multi-objective Optimization Algorithm Based on BBO for Virtual Machine Consolidation Problem." *Parallel and Distributed*

- Systems (ICPADS), 2015 IEEE 21st International Conference on IEEE, 2015.
- [11] Mi, H., et al. "Online selfreconfiguration with performance guarantee for energy-efficient largescale cloud computing data centers," in *Services Computing (SCC)*, 2010 IEEE International Conference on, July 2010, pp. 514-521.
- [12] Gao, Y., et al. "A multi-objective ant colony system algorithm for virtual machine placement in cloud computing." *Journal of Computer & System Sciences* 79.8(2013):1230-1242.
- [13] Jeyarani, R., et al. "Self adaptive particle swarm optimization for efficient virtual machine provisioning in cloud," *Int. J. Intell. Inf. Technol.*, vol. 7, no. 2, pp. 25-44, Apr. 2011.
- [14] Marotta, A., et al. "A Simulated Annealing Based Approach for Power Efficient Virtual Machines Consolidation." *Cloud Computing (CLOUD)*, 2015 IEEE 8th International Conference on IEEE, 2015.
- [15] Simon, D. "Biogeography-Based Optimization." *Evolutionary Computation IEEE Transactions on* 12.6(2009):702-713.
- [16] Simon, D, et al. "Markov Models for Biogeography-Based Optimization." *IEEE Transactions on Systems Man & Cybernetics Part B Cybernetics A Publication of the IEEE Systems Man & Cybernetics Society* 41.1(2011):299-306.
- [17] Du, D., et al. "Biogeography-based optimization combined with evolutionary strategy and immigration refusal." *Systems, Man and Cybernetics, 2009. SMC 2009. IEEE International Conference on. IEEE, 2009.*
- [18] Ma, H., et al. "Biogeography-based optimization with blended migration for constrained optimization problems." *Proceedings of the 12th annual conference on Genetic and evolutionary computation. ACM, 2010.*
- [19] Simon, D. "A dynamic system model of biogeography-based optimization." *Applied Soft Computing* 11.8(2011):5652-5661.
- [20] Rarick, R., et al. "Biogeography-based optimization and the solution of the power flow problem." *Systems, Man and Cybernetics, 2009. SMC 2009. IEEE International Conference on. IEEE, 2009.*
- [21] Ma, H. "An analysis of the equilibrium of migration models for biogeography-based optimization." *Information Sciences* 180.18(2010):3444-3464.
- [22] Ma, H., et al. "Blended biogeography-based optimization for constrained optimization." *Engineering Applications of Artificial Intelligence* 24.3(2011):517-525.

MyBatRecommender: Automated optimization of energy consumption for Android smartphones in software layer

Marcel Popolin de Araújo Cunha
UFSCar - *campus* Sorocaba
Sorocaba, São Paulo 18052-780
Email: mpopolin@gmail.com

Luciana Aparecida Martinez Zaina
UFSCar - *campus* Sorocaba
Sorocaba, São Paulo 18052-780
Email: zaina.luciana@gmail.com

Abstract—Nowadays smartphones are composed of a wide range of sensors, components and resources such as GPS (Global Positioning System), Bluetooth and Internet connection through Wi-Fi, 3G, among others. Along with the smartphone's increasing popularity around the world, there is an increasing development and popularity of power-hungry applications: applications that take advantage from these resources and may reduce the energy life time of smartphones to a few operation hours a day. The article goal is to present a mechanism that is able to dynamically manage the smartphone components states, for example turning off unnecessary interfaces, at run time. For this the mechanism collects and analyze data from the smartphone usage along the days in order to predict when the components should be managed. The experimental results show that up to 30% of energy savings is achieved when comparing to the energy dissipation of an smartphone without the proposed mechanism installed.

Index Terms—Energy management, Energy optimization, Mobile applications, Context sensitive applications.

I. INTRODUCTION

Nowadays smartphones have become extremely popular and technologically evolved, being equipped with a wide range of sensors, such as: GPS, light sensors, accelerometer, network interfaces, CPUs (Central Processing Unit) and many others. This scenario has allowed that a wide variety of applications for these smartphones have been developed. From e-mails managers to navigation systems, they are interested in providing a rich user experience, making intense use of the components available on the device and therefore consuming energy. For example, the context-sensitive applications, that are able to adapt their operations without explicit intervention of the users, providing information and services that are relevant for users to perform their tasks using information taken out of the interaction context [1]. For this, they collect user's contextual information, using sensors present on device. An example are the navigation systems, that uses the GPS to locate the user in a determinate map.

Basically, the amount of energy available on device is the result between the difference of the energy provided by the battery and the energy consumed by the sensors and components. It is considered that the energy consumption by the sensors and components is increasing and that the

technological evolution of the smartphone's battery has not followed the evolution of the other components of the device, resulting in a frequent scenario where the smartphone is out battery before the end of the day, as stated by [2]. Although there is a crescent effort on studies that aim to evolve the battery on hardware level, new ways to manage and optimize the energy consumption are necessary. Due to this lack, many researchers are concentrating their efforts to provide solutions on software layer.

The researches on this area exist before the popularization of smartphones, with some studies dated of 2004, when a similar problem occurred on PDAs (Personal Digital Assistant) [3]. However, the research on this area has increased with the smartphone's and its applications popularization, in 2011, mainly to the Android based smartphone popularization, with around 380 thousands applications and 10 billion downloads [4]. Basically, the Android framework is organized in the architectural layer represented in the Figure 1. The first layer is the layer based on the kernel Linux, which handles the access to the hardware level components via drivers. Above this layer there are other two layers of libraries: one developed in C and C++ programming languages and the other developed in the Java programming language. This last one is the responsible for providing the interface with the last layer, that is the layer of Android applications, exposing the hardware controls (e.g., managing sensor states) through APIs (Application Programming Interface) and *wakelocks*¹ to developers [5], [6]. This approach could lead to the development of energy inefficient applications if the hardware components are not used correctly, leading to a unnecessary energy waste [7].

Based on demand of reducing the energy consumption, this paper has the main goal of presenting a solution in the software layer to save energy on smartphones. This is done by presenting a mechanism, MyBatRecommender, that manages dynamically the state of the components of the smartphone. The proposed mechanism efficiency is also measured using experimental tests.

The remainder of this paper is organized as follow: the

¹developer.android.com/reference/android/os/PowerManager.WakeLock.html

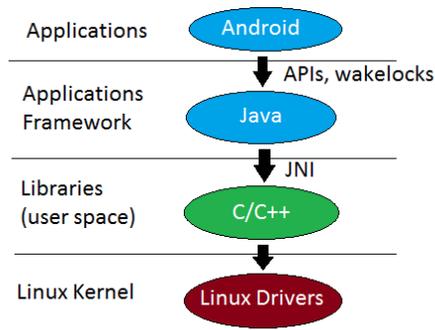


Fig. 1. Android architectural overview

Section II presents the main related works of the present study, the Section III presents the MyBatRecommender mechanism characteristics and implementation for the Android operational system. The Section IV presents the conduction and results of MyBatRecommender experimentation. Section V summarizes the results of the present study, and concludes indicating possible new directions for further works.

II. BACKGROUNDS AND RELATED WORKS

The researches with different approaches have increased in this area and could also be observed also in a qualitative way. Ones address the subject from analyzing the integration of smartphones with Cloud Computing as discussed by [8] and [9] that are focused in the Cloud offloading method, that tries to offload the execution of part of the application to the Cloud, to sensor's optimizations as discussed by [10], that tries to improve the energy efficiency of location components and others that identify how the energy is consumed on the device by its sensors and applications, as discussed by [11]. This last category of studies helps to identify new ways which could be explored to save energy and is the base for another category of studies that proposes solutions for the management and optimization of energy consumption. They are complete, complex and include most of sensors and components available on a device, as the work presented by [12] that shows how the components 3G, Wi-Fi and GPS consume energy on different states such as idle, concluding that, even in the idle state, when the components are turned on but are not actively executing their tasks, they have an energy consumption of around 25% of the overall system. Based on these measurements the authors propose a technique that try to avoid this unnecessary energy consumption.

Furthermore, recently, the company Google has shown its interest in this research area presenting in its Android new version, named Android Marshmallow, some features related to energy savings [13].

III. MYBATRECOMMENDER

The energy consumption in smartphones varies according to the user usage profile, that dictates how the components are used along the days. These components are turned on to provide the resources needed by the applications, but may

still turned on when they are not necessary any more, what consequently increases the energy consumption. Considering this, one possible approach to reduce the energy consumption is to avoid that the components stay on unnecessary states, turning them off when they are not being used. Based on this, the current work proposes a mechanism, named MyBatRecommender, which aims managing and optimizing the energy usage in smartphones to reduce it. This mechanism acts managing the states of the smartphone's components, using for this the user usage profile, that is created by the analysis of the smartphone usage data, collected along the days. The "recommender" part of the mechanism name exists considering that the MyBatRecommender, as will be further detailed, adopts its operation depending on the user, managing different sensor and components according to the user usage profile.

Figure 2 shows an overview of MyBatRecommender, that is composed by four main elements, MyBatLogger (1), MyBatServer (2), MyBatProfile (3) and MyBatSaver (4), which will be detailed on the following sections. Also, the mechanism works in a continuous way, once that each element is always executing its role to keep improving the mechanism efficiency. The mechanism is also automatic in the way that once it is installed on the user's smartphone it will automatically run and execute its role without the need of user intervention, running always in background, not interfering in the user experience with the smartphone.

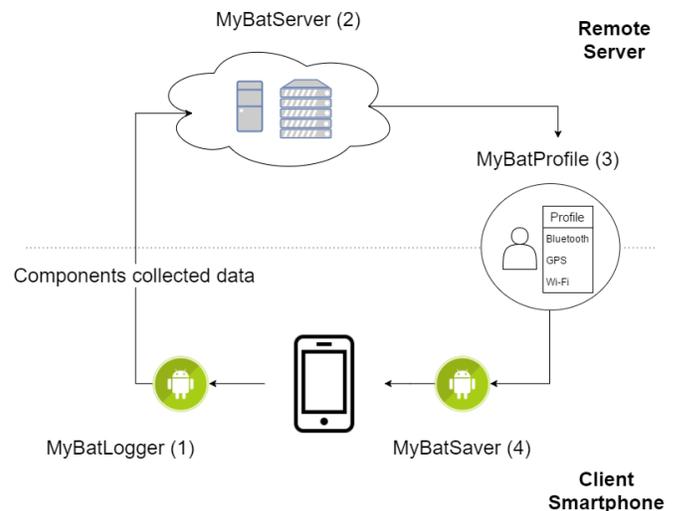


Fig. 2. MyBatRecommender overview

A. MyBatLogger

To understand how the components behave on users smartphones, data regarding the utilization of components should be gathered to be further analyzed in order to create the user usage profile. The element MyBatLogger is responsible for collecting data regarding the components *Bluetooth*, *Display*, *Wi-Fi*, *Battery*, *Mobile Network*, *GPS*, and sending to a remote server.

Based on this, the element MyBatLogger was implemented as an application for Android smartphones. This application is composed by a main Service² that is continuously running in background since the smartphone boot. This Service is the responsible for collecting the data of all above mentioned components every minute or of a specific component when there is some event regarding that component, for example, when Wi-Fi is turned on or off. This operation is denominated "generate log" where the *log* is the bundle of information representing the data collected of a component. Each component has specific data that is collected besides the *id*, unique for each smartphone, the *timestamp*, representation of when the *log* was generated and *type*, indicating the component that the *log* represents. The specific data and its collecting method depends on each component, but, in general, they represent the state of that component (on or off) and the connection state - if the component was connected, connecting, sending or receiving data, among other possible states.

B. MyBatServer

The MyBatServer element is the remote server side of the mechanism. It is the responsible for receiving requests from the client side, represented by MyBatLogger and MyBatSaver elements, and processing them according to three main actions. The first action is "Insert logs", responsible for receiving the collected *logs* from the MyBatLogger and inserting them on the corresponding database. The second action is "Create MyBatProfile", which analyses all the data collected and stored in order to generate the MyBatProfile, that represents the user usage profile. The MyBatProfile creation process will be explained on Section III-C. The last action, "Request MyBatProfile", is the responsible for returning the MyBatProfile created to the requesting part.

As the number of collected *logs* is huge, the MyBatServer element has fundamental role in storing and processing this data, once that if this task was done by the smartphone a non negligible amount of memory and energy would be consumed, decreasing the efficiency of the mechanism.

C. MyBatProfile

The element MyBatProfile represents the user usage profile, indicating how the smartphone's components are used along the days by the user. It is created by the MyBatServer, when the action "Create MyBatProfile" is invoked, and it is also the responsible for showing the MyBatSaver how it should behave, displaying the components states over the periods and days.

The MyBatRecommender's idea is to delegate the creation methods of MyBatProfile to the implementation itself, meaning that different methods can be used in order to analyze the data and generate MyBatProfile. Based on this, the current implementation is based on the existence of a user's daily routine. For example, for a certain university student, this routine could be the student going from his house to the

university, watching the lessons and at the end of the lessons, going back to his house. Then, through this daily routine it is possible to obtain a smartphone usage daily routine, because the smartphone usage depends on certain environment variables such as availability of Wi-Fi networks.

Based on this, the present paper proposes the MyBatProfile generation so it will determine the configuration of the smartphone components for each period in a day and for every day in a week. To determine how many periods a day is composed by, a questionnaire was applied to six different persons, asking about how many different environments they visit along the day, once that, as discussed above, the environments may change directly the way the smartphone is used. The result was an average of 5.4 periods. As this number must be an integer, it was considered 6 as the number of periods. Each period has a start and end time that, considering a twenty four hours day, is four hours long.

To create the MyBatProfile, five main developed algorithms are executed, resulting in the profile element created and inserted in the MyBatServer local database. They are:

- **separateByDays()**: this is the first step executed in order to create the MyBatProfile and is the responsible for grouping the collected *logs* according to the week day they were collected, using for this the attribute *timestamp*.
- **separateByPeriods()**: once the *logs* are grouped in days, the second step runs and, for each day, groups the *logs* in the six defined periods.
- **createUserProfile()**: having the *logs* grouped in days and periods, they are ready to be analyzed to create the MyBatProfile. So this step runs and, for each period, of each day, executes the last method, *analyzeType()*, which will be the responsible for determining the state of that component for that specific period.
- **analyzeType()**: this method analyzes the collected *logs* of each period to determine the state of the components for that period. The *logs* of each component are analyzed individually, using for this the attribute *type*. For this, the analysis considers, for each *log*, if it was **used or not**, condition that varies for each component, explained on Table I. Then, if the number of the *logs* considered **used** is greater than the number of *logs* considered **not used**, the state of that component, for that period, is set to be **on**, otherwise it set to be **off**.
- **insertProfile()**: finally, with the MyBatProfile created, the last step is the responsible for inserting it on the MyBatServer database so it can be fetched later.

As mentioned in the beginning of this section, MyBatRecommender is continuously executing, meaning that it is always collecting new data from the user smartphone to be analyzed and create or update the MyBatProfile. This approach tries to improve the energy efficiency seen that it keeps track of the user usage, so if user change his operating way the MyBatProfile will be updated to reflect this new operating way.

²developer.android.com/guide/components/services.html

³developer.android.com/reference/android/os/BatteryManager.html

TABLE I
CONDITIONS TO CONSIDER COMPONENTS USED OR NOT

Component	Condition
Battery	The battery state must be different than BATTERY_STATUS_CHARGING defined by BatteryManager ³
Bluetooth	The Bluetooth state must be equals to STATE_ON and the Bluetooth connection state must be equal to STATE_CONNECTED or STATE_CONNECTING, defined by BluetoothAdapter ⁴
GPS	The GPS state must be equals to GPS_EVENT_STARTED or GPS_EVENT_SATELLITE_STATUS defined by GpsStatus ⁵
Display	The Display state must be ON, represented by Intent.ACTION_SCREEN_ON, defined by Intent ⁶
Mobile networks	The Mobile networks state must be equals to DATA_CONNECTING or DATA_CONNECTED and the connection state must be one of DATA_ACTIVITY_IN, DATA_ACTIVITY_OUT or DATA_ACTIVITY_INOUT, defined by PhoneStateListener ⁷
Wi-Fi	The Wi-Fi state must be WIFI_STATE_ENABLED, defined by WifiManager ⁸ and the connection state must be CONNECTED or CONNECTING, defined by NetworkInfo.DetailedState ⁹

D. MyBatSaver

The MyBatSaver is the element responsible to apply the configuration defined by MyBatProfile. For this MyBatSaver interacts with MyBatServer in order to retrieve the last MyBatProfile created, using the smartphone unique *id* as shown in Section III-A. Once in possession of the profile, the element schedules itself to run at the start of each period, using for this the Android AlarmManager¹⁰. Then, when executed, the element retrieves the configuration linked to that period and acts changing the state of the components accordingly, turning the components on or off. After this, MyBatSaver waits for the next period to start, so it can repeat the steps and change the smartphone components states again, reflecting the new period configuration.

This operating way creates a dependence on the number of periods, once that the MyBatSaver executes only at the start of each period. Thus it is important to base the number of periods chosen on the user behavior, as explained in the beginning of this section. Also, other ways could be explored to change this dependency, as stated in section V.

⁴developer.android.com/reference/android/bluetooth/BluetoothAdapter.html

⁵developer.android.com/reference/android/location/GpsStatus.html

⁶developer.android.com/reference/android/content/Intent.html

⁷developer.android.com/reference/android/telephony/PhoneStateListener.html

⁸developer.android.com/reference/android/net/wifi/WifiManager.html

⁹developer.android.com/reference/android/net/NetworkInfo.DetailedState.html

¹⁰developer.android.com/reference/android/app/AlarmManager.html

IV. EXPERIMENTATION

In this work, the experimentation phase was carried out by running two main experiments. The first one was the measurement of the mechanism overhead and the second was the measurement of the mechanism efficiency. To run both experiments, two smartphones Sony Xperia Z2¹¹, with the same configuration, were used and they will be referenced as DUT 1 and DUT 2 in the remainder of this section, where DUT are the initials of Device Under Test, indicating that these smartphones were used to run the necessary tests.

A. MyBatRecommender overhead

The proposed mechanism, when implemented to the Android operational system, requires the usage of some smartphone's components and resources, what, consequently, incurs in energy consumption. This energy consumption, necessary to make the mechanism run, is considered the system overhead, for the current work.

To measure the MyBatRecommender overhead, the present work used the Keithley Source Measure Unit/Charge Simulator¹², an external equipment able to measure the electric current in device and, with an auxiliary software, save this data on computer for posterior analysis.

To run this experiment, the DUTs were configured in the following way: the DUT 1 had no mechanism installed and the DUT 2 had the MyBatRecommender installed. Also, both DUTs ran the experiment with the screen turned off, the airplane mode turned on and only with the component Wi-Fi turned on and connected to a network. This was necessary to avoid any energy consuming element besides the MyBatRecommender execution, that needs the Wi-Fi connection to send the collected data to the remote server. Considering this, both DUTs were connected to the Keithley Source Measure Unit during one hour to collect the data regarding the electric current. As Keithley Source Measure Unit has a sample rate of two second, the period of one hour resulted in a group of one thousand and eight hundred samples collected for each DUT, so only a sample is shown on Table II to demonstrate the format of the data.

TABLE II
KEITHLEY SOURCE MEASURE UNIT COLLECTED DATA

Time (ms)	Time	Offset (ms)	Current (mA)
1452099728914	15:02:08	531967	7
1452099730789	15:02:10	533842	7
1452099732664	15:02:12	535717	7
1452099734540	15:02:14	537593	7
1452099736415	15:02:16	539468	7

To analyze the data and compare the energy consumption of both DUTs, initially, the electric current average (AVG) was calculated for each DUT:

$$AVG = \frac{\sum(\text{electric current})}{\text{number of samples}} \quad (1)$$

¹¹http://www.sonymobile.com/global-en/products/phones/xperia-z2/

¹²http://www.tek.com/sites/tek.com/files/media/media/resources/2308.pdf

After, considering the DUTs battery capacity, 3200 mAh, it was calculated the representativity (REP), which indicates the DUT energy consumption per hour, in percentage:

$$REP = \frac{AVG * 100}{3200} \quad (2)$$

Finally, the difference of REP for both DUTs represents the overhead of the mechanism. The Table III show these values for each DUT.

TABLE III
OVERHEAD DATA ANALYSIS RESULTS

DUT	Start time	End time	MED (mA)	REP (%)
1	13:45:15	14:45:15	9.212389381	0.2878
2	15:02:08	16:02:09	10.12324493	0.3163
			Difference	0.0285

By the analysis of the data presented by the Table III, there is a difference of energy consumption between the DUTs of 0.0285% of energy per hour. Considering that the configuration difference of these DUTs is the presence of the MyBatRecommender on DUT 2, the conclusion is that the proposed mechanism has an overhead of 0.0285% of energy per hour. This value will be used on the posterior analysis to remove the mechanism overhead from the calculations when needed.

B. MyBatRecommender efficiency

The mechanism efficiency validation was driven by a proposed scenario, responsible for determining which component would be in which state for all periods in a day and all days in a week.

To run this experiment, the DUTs were configured in the following way: the DUT 1 had the MyBatRecommender installed and the DUT 2 had only the MyBatLogger element installed. This was necessary because the element MyBatLogger, present on both DUTs, was the responsible for collecting the information regarding the battery level and uploading it to the remote server, so it could be analyzed posteriorly. Considering this, both DUTs ran the experiment over two weeks, following the instructions presented by the proposed scenario. For DUT 1, both weeks were used to collect information about the battery level, but the first week was also the responsible for collecting the data necessary to create the MyBatProfile, and the second week was also the responsible for applying the energy saving actions, defined by the generated MyBatProfile. For DUT 2, as it had only the element MyBatLogger installed, both weeks were responsible only for collecting information about the battery level.

The data analysis was driven by a research question an three main hypothesis, where one is accepted rejecting the other two. The research question is: *Is there difference of energy consumption between both DUTs?* and the hypothesis are: the null hypothesis (H_0), where there is no significant ($\geq 5\%$) difference of energy consumption between the DUTs, the first alternative hypothesis (H_1), where the DUT 1, with

the MyBatRecommender installed, has a smaller significant difference of energy consumption than the DUT 2 and the second alternative hypothesis (H_2), where the DUT 2, without the mechanism installed, has a smaller significant difference of energy consumption than the DUT 1.

Based on this, to analyze and compare the energy consumption of both DUTs, the consumption rate (μ_T) was calculated for each DUT. The consumption rate is the percentage of energy consumed in a period (ϕ) divided by the time, in hours, of this period (τ). For this only the discharging periods of the experiment were considered. Discharging periods are periods where the energy is being consumed. The Tables IV and V show these values for the first and second weeks of the experiment, respectively.

TABLE IV
DUT 1 AND DUT 2 FIRST WEEK DATA

		τ	ϕ	μ_T
DUT 1	Period 1	66	94	1.4242
	Period 2	27	26	0.9629
	Period 3	54	45	0.8333
		Average		1.0734
DUT 2	Period 1	66	83	1.2575
	Period 2	27	29	1.0740
	Period 3	56	33	0.5892
		Average		0.9735

TABLE V
DUT 1 AND DUT 2 SECOND WEEK DATA

		τ	ϕ	μ_T
DUT 1	Period 1	60	35	0.5833
			Average	0.5833
DUT 2	Period 1	61	55	0.9016
			Average	0.9016

By the Table IV analysis, it is noticeable that the average of consumption rate for the first week is very similar for both DUTs, what is expected, once that in this week the MyBatRecommender was only collecting data in both devices. This is not the same result shown by the Table V where the DUT 1, with the MyBatRecommender installed, has a considerable (0.3183%) smaller average of consumption rate than the DUT 2. Discounting the overhead of the mechanism for the DUT 2, that used it only for collecting data regarding the battery level, the difference results in 0.2898%.

It is needed to validate, with some level of significance, if it is possible to reject the null hypothesis over the acceptance of one of the alternative hypothesis, considering the samples collected of both DUTs for the second week. For this, initially, the normal distribution was checked to define the statistical method to be applied and compare both samples. Using the Ryan-Joiner¹³ method through the auxiliary tool Minitab¹⁴, it was obtained that both samples have the p-value < 0.01 , meaning that they do not have the normal distribution, as the Figures 3 and 4 show.

¹³<http://www.statsref.com/HTML/index.html?ryan-joiner.html>

¹⁴<http://www.minitab.com/>

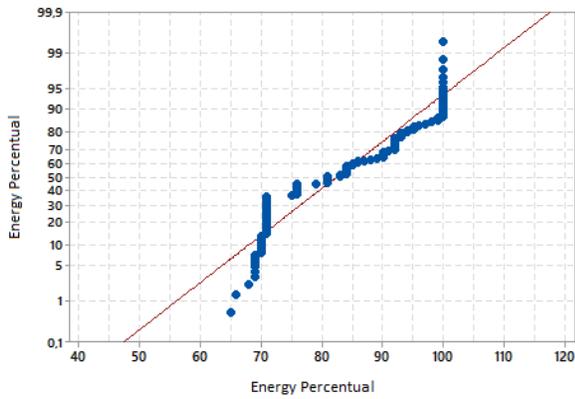


Fig. 3. Normally test for samples from the second week of DUT 1

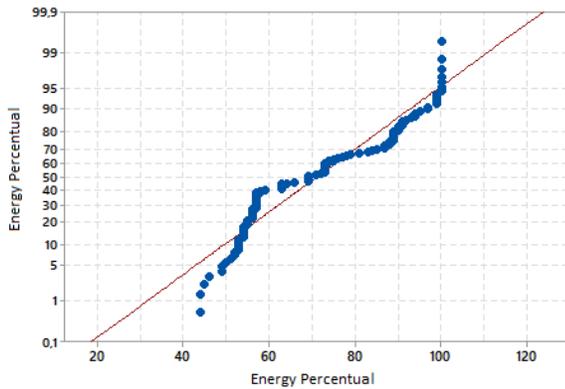


Fig. 4. Normally test for samples from the second week of DUT 2

Based on the normality test result, the Mann-Whitney U-Test¹⁵ was chosen. Considering a level of significance $\alpha=0.05$, it was obtained that the calculated p-value=0.0 is smaller than 0.05, rejecting the null hypothesis and confirming with a confidence level of 95% that the difference of energy consumption has statistical significance and it was probably caused by the presence of the mechanism MyBatRecommender.

Finally, calculating the relative difference of energy consumption between the DUTs, of 32%, the alternative hypothesis H1 is accepted, rejecting the other hypothesis. This result demonstrates that the proposed mechanism, MyBatRecommender, has an efficiency of 32% of energy savings when applied to the proposed scenario and compared to a smartphone without any mechanism installed.

V. CONCLUSION

The contributions of this work are: (i) the proposal of the MyBatRecommender, a mechanism for managing and optimizing the energy consumption in smartphones; (ii) its implementation for the Android operational system; (iii) its validation through experimentation, demonstrating the energy

savings achieved when the mechanism is used in a controlled scenario. As future works this study can be extended to implement the proposed mechanism for other smartphone operational systems, such as iOS¹⁶. It can also be improved including new components to be analyzed such as NFC (Near Field Communication) and using other algorithms to generate the MyBatProfile than the proposed by the current work, as including a learning module that defines the number and duration of the periods dynamically instead of considering them static. Finally new experiments could be elaborated and applied, such as analyzing the energy saving by components, in order to validate the outcomes.

REFERENCES

- [1] Anind K. D. and Gregory D. A., Towards a Better Understanding of Context and Context-Awareness, 1999.
- [2] Robinson, Stuart, Cellphone energy gap: Desperately seeking solutions, 2009.
- [3] Krintz, C. and Wen, Y. and Wolski, R., Application-level prediction of battery dissipation, 2004.
- [4] Google Play Wiki Page, http://en.wikipedia.org/wiki/Google_Play, 2013.
- [5] Corral, L. and Georgiev, A.B. and Sillitti, A. and Succi, G., A method for characterizing energy consumption in Android smartphones, 2013.
- [6] Pathak, Abhinav and Jindal, Abhilash and Hu, Y. Charlie and Midkiff, Samuel P., What is Keeping My Phone Awake?: Characterizing and Detecting No-sleep Energy Bugs in Smartphone Apps, 2012.
- [7] A. Pathak, A. Jindal, Y. C. Hu, and S. P. Midkiff, What is keeping my phone awake? Characterizing and detecting no-sleep energy bugs in smartphone apps, 2012.
- [8] Marin, Radu-Corneliu and Dobre, Ciprian, Reaching for the Clouds: Contextually Enhancing Smartphones for Energy Efficiency, 2013.
- [9] Fekete, K. and Csorba, K. and Forstner, B. and Vajk, T. and Feher, M. and Albert, I., Analyzing computation offloading energy-efficiency measurements, 2013.
- [10] Oshin, T.O. and Poslad, S. and Ma, A., Improving the Energy-Efficiency of GPS Based Location Sensing Smartphone Applications, 2012.
- [11] Vallina-Rodriguez, N. and Crowcroft, J., Energy Management Techniques in Modern Mobile Handsets, 2013.
- [12] Donohoo, B.K. and Ohlsen, C. and Pasricha, S. and Yi Xiang and Anderson, C., Context-Aware Energy Enhancements for Smart Mobile Devices, 2014.
- [13] Google IO, <https://www.android.com/intl/en/versions/marshmallow-6-0/>, 2015.

¹⁵<https://statistics.laerd.com/minitab-tutorials/mann-whitney-u-test-using-minitab.php>

¹⁶<http://www.apple.com/ios/>

A Multi-Source TrAdaBoost Approach for Cross-Company Defect Prediction

Xiao Yu¹, Jin Liu^{1*}, Mandi Fu^{2,3}, Chuanxiang Ma^{2,3*}, Guoping Nie⁴, Xu Chen¹

¹State Key Lab. of Software Engineering, Computer School, Wuhan University, Wuhan, China

²School of Computer Science and Information Engineering, HuBei University, Wuhan, China

³Educational Informationalization Engineering Research Center of HuBei Province, Wuhan, China

⁴School of Mathematics, Huazhong University of Science and Technology, Wuhan, China

*Corresponding author email: jinliu@whu.edu.cn, mx838@hubei.com

Abstract—Cross-company defect prediction (CCDP) is a practical way that trains a prediction model by exploiting one or multiple projects of a source company and then applies the model to target company. Unfortunately, larger irrelevant cross-company (CC) data usually makes it difficult to build a prediction model with high performance. On the other hand, brute force leveraging of CC data poorly related to within-company (WC) data may decrease the prediction model performance. To address such issues, this paper introduces Multi-Source TrAdaBoost algorithm, an effective transfer learning approach to perform CCDP. The core idea of our approach is that: 1) employ limited amount of labeled WC data to weaken the impact of irrelevant CC data; 2) import knowledge not from one but from multiple sources to avoid negative transfer. The experimental results indicate that: 1) our proposed approach achieves the best overall performance among all tested CCDP approaches; 2) only 10% labeled WC data is enough to achieve good performance of CCDP by using our proposed approach.

Keywords—software defect prediction; cross-company defect prediction; transfer learning; Multi-Source TrAdaBoost

I. INTRODUCTION

Software defect prediction is one of the most important software quality assurance techniques. It aims to detect the defect proneness of new software modules via learning from defect data. So far, many efficient software defect prediction approaches [1-6] have been proposed, but they are usually confined to within company defect prediction (WCDP). WCDP works well if sufficient data is available to train a defect prediction model. However, it is difficult for a new company to perform WCDP if there is limited historical data. Cross-company defect prediction (CCDP) is a practical approach to solve the problem. It trains a prediction model by exploiting one or multiple projects of a source company and then applies the model to target company [7].

Most existing CCDP approaches [7-14] focus on using only cross-company (CC) data to build a proper prediction model. Unfortunately, larger irrelevant CC data usually makes it difficult to build a prediction model with high performance [15]. In fact, if there is limited amount of labeled WC data, the data is not enough to perform WCDP, but it may help a lot to improve the performance of CCDP. Another scenario is that companies may already have their defect prediction models in

place and making use of CC data may improve the performance of models [16].

The challenges of performing CCDP with limited amount of labeled WC data usually include:

1) How to weaken the impact of irrelevant CC data to improve the performance of CCDP.

The ability to transfer knowledge from a source company to a target company depends on how they are related. The stronger the relationship, the more usable will be CC data. The performance of CCDP is generally poor because of larger irrelevant CC data. The irrelevant data has bad effects on the prediction outcome [17].

2) How to avoid negative transfer when leveraging multiple cross-companies data.

Brute force leveraging of CC data poorly related to WC data may decrease the prediction model performance. Lin et al. developed the double transfer boosting (DTB) approach [15] for CCDP. DTB approach merges all CC data as a source, relies on only the source, and therefore is intrinsically vulnerable to negative transfer.

Considering the above challenges, this paper introduces Multi-Source TrAdaBoost algorithm [17], an effective transfer learning approach to perform CCDP. We call the proposed approach for CCDP as MStrA. The core insight of MStrA is that: 1) in order to narrow the distribution gap between CC data and WC data, MStrA firstly uses NN filter [10] to select the k most similar CC data to each WC data and then use data gravitation [18] for reweighting the whole distribution of CC data to fit WC data; 2) MStrA then trains and combines a set of weak prediction models for building a stronger ensemble defect prediction model using not only reweighted CC data but also limited amount of WC data; 3) in each training round, MStrA transfers knowledge from multiple sources and reduces the weights of irrelevant CC data continuously.

To assess the MStrA approach, this paper explores the following research questions.

RQ1: How effective is our proposed MStrA approach when comparing to other approaches for CCDP?

RQ2: How much labeled WC data is enough to help the prediction model achieve better performance by using our proposed MStrA approach?

The remainder of this paper is organized as follows. Section 2 presents the related work. Section 3 describes the proposed MStrA approach for CCDP. Section 4 demonstrates the experimental results. Finally, Section 5 addresses the conclusion and points out the future work.

II. RELATED WORK

In this section, we briefly review the existing cross-company and cross-project defect prediction approaches. These approaches can be categorized into two main types: defect prediction using only CC data [7-14], defect prediction using not only CC data but also limited amount of labeled WC data [15-16].

A. Defect prediction using only CC data

In order to solve the problem that the new companies have too limited historical data to perform WCDP well, the cross-project and cross-company defect prediction appeared.

Briand et al. [8] used logistic regression and MARS models to learn a defect predictor, which is also the earliest work on CCDP. Zimmermann et al. [9] studied CCDP models on 12 real-world applications datasets. Their results indicate that CCDP is still a serious challenge. Turhan et al. [10] investigated the applicability of CC data for building localized defect predictors using 10 projects collected from two different companies including NASA and SOFTLAB. And they have proposed a nearest neighbor (NN) filter to select CC data. He et al. [11] investigates defect predictions in the cross-project context focusing on the selection of training data. Furthermore, they proposed an approach to automatically select suitable training data for projects without historical data so that the results of their experiments are comparable with WCDP, which indicated that some approach of CCDP can be comparable to WCDP. They noted that learning predictors using the data from other projects can be a potential way to defect prediction without any historical data. In order to find data for quality prediction, Peters et al. [12] introduced the Peters filter to select training data via the structure of other projects. They compared the filter with two other approaches for quality prediction to assess the performance of the Peters filter, and found that 1) WCDP are weak for small data sets; 2) the Peters filter + CCDP builds better and more useful predictors. Zhang et al. [13] proposed sample-based methods for software defect prediction. For a large software system, they could select and test a small percentage of modules, and then built a defect prediction model to predict defect-proneness of the rest of the modules. They described three methods for selecting a sample and proposed a novel active semi-supervised learning method ACoForest to facilitate the active sampling. The results showed that the proposed methods are effective and have potential to be applied to industrial practice. Ma et al. [14] proposed a novel algorithm called Transfer Naive Bayes (TNB) to transfer cross-company data information into the weights of the training data and then build the predictor based on re-weighted CC data. The results indicated that TNB is more accurate in terms of

AUC, within less runtime than the state of the art methods and can effectively achieve the CCDP task. The heterogeneous CCDP (HCCDP) task is that the source and target company data is heterogeneous. Jing et al. [7] provided an effective solution for HCCDP. They proposed a unified metric representation (UMR) for the data of source and target companies and introduced canonical correlation analysis (CCA), an effective transfer learning method, into CCDP to make the data distributions of source and target companies similar. Results showed that their approach significantly outperforms state-of-the-art CCDP methods for HCCDP with partially different metrics and for HCCDP with totally different metrics, their approach is also effective.

The approaches above are focus on using only CC data to build predictors. Considering there is limited amount of labeled WC data, the data is not enough to perform with company defect prediction, but it may help a lot to improve the performance of CCDP.

B. Defect prediction with limited amount of labeled WC data

Turhan et al. [16] introduced a mixed model of within and cross data for CCDP to investigate the merits of using mixed project data for binary defect prediction. Results showed that when there is limited project history, mixed model for CCDP can achieve good performance which can be comparable to WCDP. It provided a new idea to CCDP that the use of a small amount of labeled WC data would be very valuable to improve the performance of CCDP.

Lin et al. [15] introduced a novel approach named Double Transfer Boosting (DTB) to narrow the gap of different distributions between CC data and WC data and to improve the performance of CCDP by reducing negative samples in CC data. However, it merges all CC data as one source and the result only relies on the single source so that it is prone to negative transfer, which is exactly what we will solve in this paper.

III. METHODOLOGY

In this section, we present our MStrA approach for CCDP. Its main steps are as follows: 1) in order to narrow the distribution gap between CC data and WC data, MStrA first uses NN filter [10] to select the k most similar CC data to each WC data and then uses data gravitation [18] for reweighting the whole distribution of CC data to fit WC data; 2) MStrA mixes limited amount of labeled WC data with reweighted CC data to build the prediction model by using Multi-Source TrAdaBoost algorithm.

A. Data Preprocessing

Previous work [10] found that using raw CC data directly would increase false alarm rates due to irrelevant instance in CC data, so several data preprocessing works should be done before building the prediction model. To decrease the negative effect of the irrelevant instance in CC data for building the prediction model, we employ NN filter proposed by Turhan et al. [10] to form the training set. Based on the widely used classification method KNN algorithm, NN filter can find out the most similar $K \times N$ instances from CC data while N is the

number of WC instances and K is the parameter of the KNN method. Note that duplicate instances may exist in this filtered dataset, as some instances in WC data may have some common neighbors in CC data. Thus, the final filtered CC training data can be formed by using only unique ones.

Then we apply the data gravitation method [18] to change the entire distribution of CC data. Suppose that an instance x_i can be described by $x_i = \{a_{i1}, a_{i2}, \dots, a_{ik}\}$, where a_{ij} is the j -th attribute value of the i -th instance and k is the number of the attributes.

(1) We compute two vectors, $\text{Max} = \{\max_1, \max_2, \dots, \max_k\}$ and $\text{Min} = \{\min_1, \min_2, \dots, \min_k\}$ to represent the attribute value distribution of WC data, where \max_i is the maximum value of the i -th attribute, \min_i is the minimal value of the i -th attribute.

(2) For each instance x_i in CC data, the degree s_i of similarity to WC data is computed according to Eq.(1)

$$s_i = \sum_{j=1}^k h(a_{ij}) \quad (1)$$

where a_{ij} is the j -th attribute value of the instance x_i , $h(a_{ij}) = 1$, if $\min_j \leq a_{ij} \leq \max_j$; otherwise, $h(a_{ij}) = 0$.

(3) The weight w_i of instance x_i in CC data can be calculated by Eq.(2) according to the formulation of data gravitation [18].

$$w_i = s_i / (k - s_i + 1)^2 \quad (2)$$

where k is the number of the attributes.

According to this formula, the weight w_i of instance x_i shows the similarity of x_i to WC data, and the greatest w_i will be assigned when $s_i = k$.

Though the above steps, the entire distribution of CC data is reweighted to be close to WC data.

B. Multi-Source TrAdaBoost Approach

Let $D^{\text{SK}} = \{(x_1^{\text{S}1}, c_1^{\text{S}1}), \dots, (x_n^{\text{S}K}, c_n^{\text{S}K})\}$ be the k -th cross-company data, where n is the number of instances in the k -th cross company data, $c_i^{\text{S}K} \in \{\text{true}, \text{false}\}$ is the class label of instance $x_i^{\text{S}K}$. Let $D^{\text{T}} = \{(x_1^{\text{T}}, c_1^{\text{T}}), \dots, (x_m^{\text{T}}, c_m^{\text{T}})\}$ be limited amount of labeled WC data, where m is the number of instances in labeled WC data, c_i^{T} is the class label of instance x_i^{T} . During NN filter and data gravitation, filtered CC data $D_{\text{S}1}, \dots, D_{\text{S}N}$ and labeled WC data D_{T} are assigned different weight according Eq.(2).

In each training round, combine the k -th cross-company data and labeled WC data to train a candidate weak prediction model. In our paper, we choose Naïve Bayes [19] as the base prediction model due to its effectiveness in defects prediction [20]. The final weak prediction model $f_t(x)$ in t -th iteration is one of the candidate weak prediction models which has the minimal prediction error on labeled WC data. In other words, every weak prediction model is selected from CC data that appears to be the most closely related to WC data. The prediction error function is defined according Eq.(3).

$$\varepsilon_t = \sum_{i=1}^m \frac{w_i^t |f_t(x_i) - c_i|}{\sum_{i=1}^m w_i^t} \quad (3)$$

$$\text{Set } \beta_t = \frac{1}{2} \ln \frac{1 - \varepsilon_t}{\varepsilon_t} \quad (4)$$

In this way, we import knowledge not from one but from multiple sources, thus decreasing the risk for negative transfer. At t -th iteration, the instances in WC data are given more importance if the instances are misclassified. They are believed to be the ‘‘most informative’’ for the next round, so the weight of the misclassified instances are increased according Eq.(5).

$$w_i^T = w_i^T e^{\beta_t |f_t(x_i^T) - c_i^T|} \quad (5)$$

The instances in CC data are given less importance if the instances are misclassified. They are believed to be the most dissimilar to WC data, so the weight of the misclassified instances are decreased according Eq.(6) in order to weaken their impacts in the next round through multiplying the Hedge(β) defined in Eq.(7).

$$w_i^{\text{SK}} = w_i^{\text{SK}} e^{-\beta_s |f_t(x_i^{\text{SK}}) - c_i^{\text{SK}}|} \quad (6)$$

$$\beta_s = \frac{1}{2} \ln \left(1 + \sqrt{2 \ln \frac{n_s}{M}} \right) \quad (7)$$

After several iterations, the instances in CC data that fit WC data will have larger training weights, while the instances in CC data that are dissimilar to WC data will have lower weights. The instances in CC data with larger training weights intend to build a better prediction model. The final prediction model $F(x)$ can be expressed as follows:

$$F(x) = \text{sign}(\sum_t \beta_t f_t(x)) \quad (8)$$

Algorithm 1 presents the pseudo-code of MSTRa approach to perform CCDP.

Algorithm 1. MSTRa approach

Input: filtered CC data $D^{\text{S}1}, \dots, D^{\text{S}N}$, limited amount of labeled WC data D^{T} , and the maximum number of iterations M

Output: a prediction model $F(x)$

1. Initialize a weight vector ($\mathbf{w}^{\text{S}1}, \dots, \mathbf{w}^{\text{S}N}, \mathbf{w}^{\text{T}}$) using Eq.(2)
 2. **for** $t=1, \dots, M$ **do**
 3. Empty the set of candidate weak prediction models
 4. Normalize to 1 the weight vector ($\mathbf{w}^{\text{S}1}, \dots, \mathbf{w}^{\text{S}N}, \mathbf{w}^{\text{T}}$)
 5. **for** $k=1, \dots, N$ **do**
 6. Train the candidate weak prediction model $f_t^{\text{K}}(x)$ over the combined data $D^{\text{SK}} \cup D^{\text{T}}$, using weight ($\mathbf{w}^{\text{SK}}, \mathbf{w}^{\text{T}}$)
 7. Compute the error of $f_t^{\text{K}}(x)$ on D^{T} using Eq.(3)
 8. **end for**
 9. Find the weak prediction model $f_t(x)$ which has the minimal error
 10. Update weights vector ($\mathbf{w}^{\text{S}1}, \dots, \mathbf{w}^{\text{S}N}, \mathbf{w}^{\text{T}}$) for the next round using Eq.(5) and Eq.(6)
 11. **end for**
 12. **return** $F(x) = \text{sign}(\sum_t \beta_t f_t(x))$
-

IV. EXPERIMENTS

In this section, we evaluate our proposed MSTRa approach to perform CCDP empirically. We first introduce the experiment dataset and the performance measures. Then, in order to investigate the performance of MSTRa, we perform some empirical experiments to find answers to the research questions mentioned above.

A. Data set

In this experiment, we employ 15 available and commonly used datasets which can be obtained from PROMISE [21]. The 15 datasets have the same 20 attributes, so we can apply all attribute information directly. Table I tabulates the details about the datasets.

TABLE I. DETAILS OF EXPERIMENT DATASET

Project	Examples	%Defective	Description
ant	125	16	Open-source
arc	234	11.5	Academic
camel	339	3.8	Open-source
elearn	64	7.8	Academic
jedit	272	33.1	Open-source
log4j	135	25.2	Open-source
lucene	195	46.7	Open-source
poi	237	59.5	Open-source
prop	660	10	Proprietary
redaktor	176	15.3	Academic
synapse	157	10.2	Open-source
systemdata	65	13.8	Open-source
tomcat	858	9	Open-source
xalan	723	15.2	Open-source
xerces	162	47.5	Open-source

B. Performance measures

In the experiment, we employ three commonly used performance measures including pd , pf and g -measure. They are defined in Table 2 and summarized as follows.

TABLE II. PERFORMANCE MEASURES

		Actual	
		yes	no
Predicted	yes	TP	FP
	no	FN	TN
pd	$\frac{TP}{TP + FN}$		
pf	$\frac{FP}{FP + TN}$		
g -measure	$\frac{2 * pd * (1 - pf)}{pd + (1 - pf)}$		

- Probability of detection or pd is the measure of defective modules that are correctly predicted within the defective class. The higher the pd , the fewer the false negative results.

- Probability of false alarm or pf is the measure of non-defective modules that are incorrectly predicted within the non-defective class. Unlike pd , the lower the pf value, the better the results.

- g -measure is a trade-off measure that balances the performance between pd and pf . A good prediction model

should have high pd and low pf , and thus leading to a high g -measure.

C. Results for Q1

In order to confirm whether the MSTRa approach can perform better than other CCDP approaches, we compared our approach with four state-of-the-art CCDP approaches. More details are provided below:

- **NN filter** [10] is based on the widely used classification method K-Nearest Neighbors (KNN) algorithm to filter irrelevant CC data. It can find out the most similar $K \times N$ instances from CC data while N is the number of instances in WC data and K is the parameter of the KNN method. In our experiment, we choose K as 10. After NN filter, Naïve Bayes classifier is chosen as the basic prediction model.

- **TNB** [14] first reweights the CC data by the data gravitation method, then builds a transfer Naïve Bayes classifier on reweighted CC data.

- **NN+WC** (Nearest-Neighbor filter with WC data) [16] mixes $p\%$ WC data with CC data which was processed by NN filter as training data. In our experiment, we choose p as 10. Then Naïve Bayes classifier is chosen as the basic prediction model on the training data.

- **DTB** [15] first uses NN filter, SMOTE [22] and data gravitation to process CC data. Then limited amount of labeled WC data and reweighted CC data are mixed to build prediction model using the transfer boosting algorithm.

In every experiment, one dataset is selected as WC data and the rest are regarded as CC data to conduct the experiment. The CC data is considered as basic training data which will be adjusted in every experiment. WC data will be randomly divided into two parts: 10% labeled WC data as training data with CC data in our MSTRa approach, DTB approach and NN+WC approach, and the remainder is taken as test data for all CCDP approaches in order to be fair. Then the values of the performance measures for our MSTRa approach, DTB approach and NN+WC approach are calculated.

The comparison results are summarized in Table 3 with three performance measures mentioned above. It shows that the NN approach often achieves the best pd but the worst pf so that it usually ends up with low g -measure value. The performance of NN+WC approach seems sometimes have lower pf than the NN approach but mostly have similar result with NN approach. The effect of WC data seems not very obvious.

It's very clear that the transfer learning models including TNB, DTB and MSTRa have lower pf than the other two models. In the aspect of pf value, the MSTRa approach reduces the pf to a large extent. The pf results of 7 projects are better than others. On more than half tests, MSTRa achieves higher g -measure than other models.

In total, the MSTRa approach has acceptable pd value and can obtain better pf values in most experiments we conducted, and it almost always achieve the higher g -measure value than other models. In other words, the MSTRa approach outperforms other CCDP approaches, therefore it can be an effective approach for CCDP.

TABLE III. PD,PF AND G-MEASURE VALUES, THE RESULTS OF FIVE APPROACH

No.	Test data	MSTrA			DTB			TNB			NN			NN+WC		
		PD	PF	G	PD	PF	G	PD	PF	G	PD	PF	G	PD	PF	G
1	ant	0.823	0.313	0.749	0.811	0.370	0.709	0.819	0.524	0.602	0.371	0.270	0.492	0.399	0.275	0.099
2	arc	0.409	0.106	0.561	0.605	0.272	0.661	0.745	0.413	0.655	0.745	0.629	0.495	0.807	0.648	0.488
3	camel	0.417	0.066	0.576	0.487	0.300	0.574	0.564	0.290	0.629	0.784	0.709	0.424	0.784	0.710	0.423
4	elearn	0.75	0.283	0.733	0.675	0.243	0.713	1.000	0.393	0.756	0.900	0.383	0.724	0.900	0.383	0.724
5	jedit	0.646	0.181	0.722	0.568	0.237	0.651	0.475	0.168	0.605	0.932	0.616	0.544	0.932	0.628	0.532
6	log4j	0.576	0.247	0.652	0.611	0.234	0.679	0.635	0.134	0.733	0.936	0.706	0.444	0.936	0.709	0.444
7	lucene	0.722	0.474	0.608	0.581	0.382	0.598	0.580	0.221	0.665	0.762	0.554	0.563	0.750	0.548	0.564
8	poi	0.551	0.241	0.638	0.632	0.415	0.607	0.416	0.228	0.540	0.910	0.678	0.475	0.910	0.684	0.469
9	prop-6	0.655	0.299	0.678	0.670	0.331	0.669	0.529	0.336	0.589	0.860	0.635	0.512	0.860	0.628	0.519
10	redactor	0.591	0.336	0.625	0.616	0.679	0.422	0.634	0.513	0.550	1.000	0.899	0.184	1.000	0.891	0.196
11	synapse	0.786	0.285	0.748	0.871	0.490	0.643	0.775	0.422	0.662	0.935	0.777	0.360	0.935	0.781	0.355
12	system	0.75	0.490	0.607	0.717	0.340	0.687	0.563	0.260	0.640	0.817	0.341	0.730	0.817	0.341	0.730
13	tomcat	0.418	0.163	0.558	0.712	0.396	0.653	0.914	0.592	0.564	0.632	0.380	0.614	0.690	0.359	0.660
14	xalan	0.458	0.175	0.589	0.654	0.400	0.625	0.604	0.356	0.623	0.961	0.679	0.481	0.961	0.685	0.474
15	xerces	0.586	0.392	0.595	0.370	0.274	0.490	0.319	0.268	0.444	0.437	0.631	0.400	0.437	0.631	0.400
	Average	0.609	0.270	0.643	0.639	0.358	0.625	0.638	0.341	0.617	0.799	0.592	0.496	0.808	0.593	0.472

D. Results for Q2

In order to confirm how much labeled WC data is enough to help the prediction model achieve better performance by using our proposed MSTrA approach, we randomly select $p\%$ (10%,15%,20%,25%,30%) WC data as training data with CC data, and the remainder data is taken as test data. In every experiment, one dataset is selected as WC data and the rest are regarded as CC data. We repeated our proposed MSTrA approach 20 times in every experiment to avoid sample bias. Then the mean values of g -measure for MSTrA are recorded in Figure 1.

As shown in Figure 1, using only 10% labeled WC data with CC data is enough to achieve good performance by using our MSTrA approach. In particular, better performance is achieved on ant, elearn, jedit and xalan datasets when using only 10% labeled WC data. There is no significant improvement on arc, synapse, system, xerces, tomcat, lucene, poi and prop6 datasets when incrementally adding labeled WC data. In total, we only need limited amount of labeled WC data (i.e. 10% is enough) to achieve good performance of CCDP by using our proposed MSTrA approach. Therefore, a new company can exploit our proposed MSTrA approach to perform CCDP at the early stages of development activities if there is limited historical data.

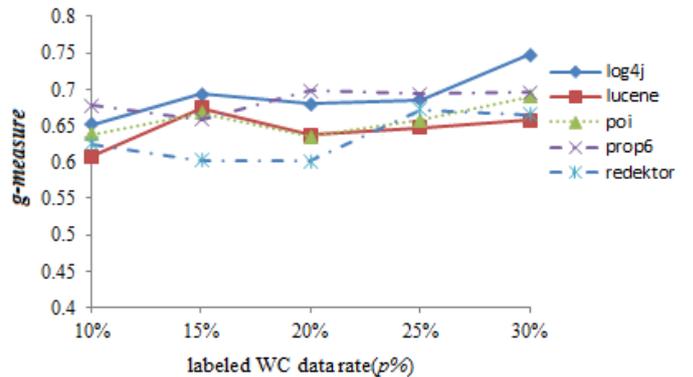
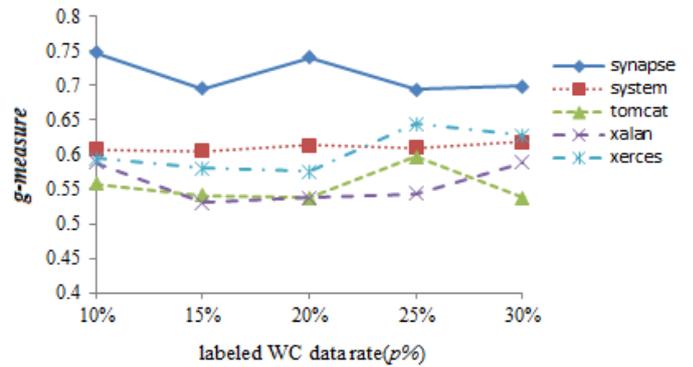
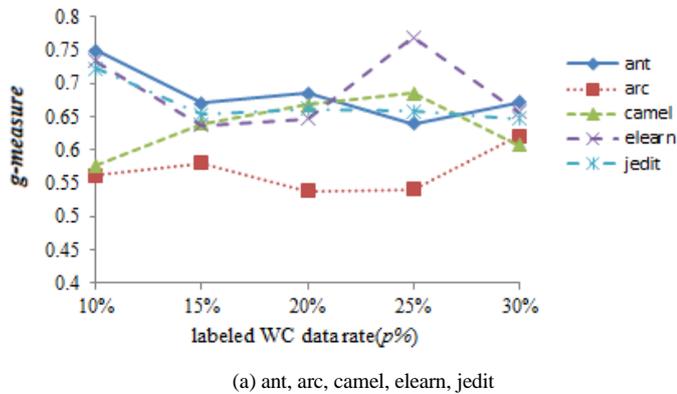


Fig. 1. G-measure performances with different size of labeled WC data

V. CONCLUSION AND FUTURE WORK

In this paper, we address the issues of how to weaken the impact of irrelevant CC data and how to avoid negative transfer when leveraging multiple source companies data to improve the performance of CCDP. We introduce Multi-Source TrAdaBoost algorithm to improve the performance of CCDP. First of all, we use NN-filter and data gravitation for reweighting the whole distribution of CC data to fit WC data.

Then we train and combine a set of weak prediction models for building a stronger ensemble defect prediction model using not only reweighted CC data but also limited amount of labeled WC data. In each training round, we transfer knowledge from multiple sources to avoid negative transfer and reduce the weights of irrelevant instances in CC data to weaken the impact of irrelevant CC data continuously.

We conduct experiments on the 15 datasets to evaluate the performance of the proposed approach. The experimental results indicate that the proposed approach can effectively weaken the impact of irrelevant data and avoid negative transfer to improve the performance of CCDP. The proposed MStrA approach is an effective approach for CCDP.

In the future, we would like to validate the generalization ability of our approach on more company data.

ACKNOWLEDGMENT

This work is partly supported by the grants of National high technology research and development program (863 Program 2012AA011204-01), National Natural Science Foundation of China (61070013, 61300042, U1135005, 71401128), the Fundamental Research Funds for the Central Universities (No. 2042014kf0272, No. 2014211020201) and Natural Science Foundation of HuBei (2011CDB072).

REFERENCES

- [1] K. Elish and M. Elish, "Predicting defect-prone software modules using support vector machines," *Journal of Systems and Software*, 2008, 81(5):649-660.
- [2] J. Zheng, "Cost-sensitive boosting neural networks for software defect prediction," *Expert Systems with Applications*, 2010, 37(6):4537-4543.
- [3] Z. B. Sun, Q. B. Song, and X. Y. Zhu, "Using coding based ensemble learning to improve software defect prediction," *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, 2012, 42(6):1806-1817.
- [4] S. Wang and X. Yao, "Using class imbalance learning for software defect prediction," *IEEE Transactions on Reliability*, 2013, 62(2):434-443.
- [5] M. Liu, L. Miao, and D. Zhang, "Two-stage cost-sensitive learning for software defect prediction," *IEEE Transactions on Reliability*, 2014, 63(2):676-686.
- [6] X. Y. Jing, S. Ying, Z. W. Zhang, S. S. Wu, and J. Liu, "Dictionary learning based software defect prediction," in: *Proc. of the 36th International Conference on Software Engineering (ICSE)*, 2014, pp. 414-423.
- [7] Xiaoyuan Jing et al, "Heterogeneous Cross-Company Defect Prediction by Unified Metric Representation and CCA-Based Transfer Learning," in: *Proc. of the 10th Joint Meeting on Foundations of Software Engineering*, 2015, pp 496-507.
- [8] Briand L C, Melo W L, Wust J, "Assessing the applicability of fault-proneness models across object-oriented software projects," *IEEE Transactions on Software Engineering*, 2002, 28(7): 706-720.
- [9] T. Zimmermann, N. Nagappan, H. Gall, E. Giger, B. Murphy, "Cross-project defect prediction: a large scale experiment on data vs. domain vs. process," in: *Proc. of the 7th Joint Meeting of the European Software Engineering Conference and the ACM SIGSOFT Symposium on the Foundations of Software Engineering*, ACM, 2009, pp. 91-100.
- [10] B. Turhan, T. Menzies, A.B. Bener, J. Di Stefano, "On the relative value of cross company and within-company data for defect prediction," *Empirical Softw. Eng.* 2009, 14 (5): 540-578.
- [11] Z. He, F. Shu, Y. Yang, M. Li, Q. Wang, "An investigation on the feasibility of cross-project defect prediction," *Autom. Softw. Eng.* 2012, 19 (2) 167-199.
- [12] Peters F, Menzies T, Marcus A, "Better cross company defect prediction," In: *Proc. of the 10th International Workshop on Mining Software Repositories*, San Francisco, CA, 2013, 409-418.
- [13] Zhang F, Mockus A, Keivanloo I, et al, "Towards Building a Universal Defect Prediction Model," In: *Proc. of the 11th Working Conference on Mining Software Repositories*, 2014, 182-191.
- [14] Y. Ma, G. Luo, X. Zeng, A. Chen, "Transfer learning for cross-company software defect prediction," *Inform. Softw. Technol.* 2012, 54 (3): 248-256.
- [15] Chen L, Fang B, Shang Z, et al. "Negative samples reduction in cross-company software defects prediction," *Information and Software Technology*, 2015, 62: 67-77.
- [16] B. Turhan, A. Tosun Mısırlı, A. Bener, "Empirical evaluation of the effects of mixed project data on learning defect predictors," *Inform. Softw. Technol.* 2013, 55 (6):1101-1118.
- [17] Yao Y, Doretto G, "Boosting for transfer learning with multiple sources," in: *2010 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2010: 1855-1862.
- [18] L. Peng, B. Yang, Y. Chen, A. Abraham, "Data gravitation based classification," *Inform. Sci.* 2009, 179 (6): 809-819.
- [19] D.D. Lewis, "Naive (Bayes) at forty: the independence assumption in information retrieval" *Machine Learning: ECML-98*, Springer, 1998, pp. 4-15.
- [20] T. Hall, S. Beecham, D. Bowes, D. Gray, S. Counsell, "A systematic literature review on fault prediction performance in software engineering," *IEEE Trans. Softw. Eng.* 2012, 38 (6): 1276-1304.
- [21] G. Boetticher, T. Menzies, T. Ostrand, The PROMISE Repository of Empirical Software Engineering Data, 2007 <<http://promisedata.org/repository>>.
- [22] N.V. Chawla, K.W. Bowyer, L.O. Hall, W.P. Kegelmeyer, "Smote: synthetic minority over-sampling technique," *J. Artif. Intell. Res.* 2002, 16 :321-357

Exploring the Influence of Time Factor in Bug Report Prioritization

Zhengjie Xu, Tieke He, Weiqiang Zhang, Yabin Wang, Jia Liu*, Zhenyu Chen
State Key Laboratory for Novel Software Technology, Nanjing University, Nanjing, China
*liujia@nju.edu.cn

Abstract

Time factor has been widely applied into a wide range of data mining areas, such as social network and information retrieval. The main idea of taking time factor into consideration is that human activities may have some relations to time pattern. However, little attention has been pulled on the study of time factor in the area of software engineering. In this paper, we endeavour to explore to what extent time factor affects the prioritization of bug reports, a specified while important task in software engineering. Specifically, we test four time factors that may have some influence on this task, which are time of day, normal time, day of week, and days to major version. After the validation of relatedness of all these factors, we conduct an extensive set of experiments on two datasets to verify the effectiveness of these factors. The experimental results demonstrate that we can effectively improve the results by metrics of both Precision and Recall, on two classical models, i.e., the SVM model and Naive Bayes model.

Keywords: Time factor; bug report prioritization; SVM

1. Introduction

Error or so-called bug reports are quite common during the process of software development and maintenance, which are critical to the stability as well as security of the software. Users, developers or any members who will use the software may submit a bug report. Thus, it is quite important for the developers to judge and analyze the bugs. A bug repository is then created to store great numbers of bug reports. Bugzilla, a bug repository came out during the development of Mozilla, is now widely used in most open-source software development. However, without being analyzed, bug reports are just none of use for the developers. Therefore, analyzing whether the bugs are valid or not, correct or not, important or not even unique or not becomes a necessary process. This is so-called bug triaging. For a bug triager, if he analyzes the bug reports one

by one in time sequence, he will be overwhelmed by lots of data and thus, some important bugs may be ignored, which does great harm to the whole maintenance of the software. Hence, a way to prioritize the bug reports becomes an increasingly need for the developers.

In most researches in prioritization, adopted factors such as the textual content of the bug report, the author and so on are quite common. For example, the DRONE [12] framework takes textual, author, related-report, severity and product as their judging features. By judging these features, we can prioritize the bug reports to some extent. Kremenek et al. [9] used the successful as well as the failed checks found by the bug-finding tools to prioritize the bugs. Tools' analyzing decisions and Z-ranking scheme are used to rank the bugs' priority. These factors are basic and fundamental which are used widely in bug prioritization.

However, there is a sort of factors that most researchers did not notice or not pay too much attention to in the area of software engineering. It is the time factor, which includes the time the bug report is handed in as well as the existing time of the bug report. All the researches tend to ignore the factor of time, which, we believe, is cursory. Other subjects have given us the lesson that time factor can be critical and should be put emphasis on: in economy, time influences the value of money; in the subject of journalism and communication, time also affects the value of the news. As we can see, time factor is critical in many areas and therefore, a potential feature in deciding the priority of the bug report. Inspired from these thoughts, we have raised the question whether time factor has an impact on the bug prioritization.

On the other hand, time-based bug prioritization is difficult to accomplish. There are lots of challenges we have to face. Unlike other factors, such as author, textual content and so on, time factor is not as clear as those features and not so easy to extract from the bug report. Meanwhile, as almost none of the current prioritization model includes time factor, how to merge the time factor into the model becomes a problem. These are two of the basic problems we meet.

To solve this problem, we first adopt significance test in judging the relevance between time factor and bug reports' priority. This part of experiment will show the relevance

roughly, which determines whether time will affect the priority of bug reports or not. Afterwards, we have made a contrast experiment on to what extents time factor can influence the priority. A constructed model in past researches will be used in the experiment as well as the model merged with the factor of time. By this means, we can easily tell the effects of merging time factor into a prioritization model.

Briefly, we process the time factors as follows. 1) For the time of day (*TOD*) factor, we divide a weekday into 24 parts by hour, and fit every report into them accordingly. 2). For the normal time (*NT*) factor, we first aggregate every reporters' reporting time during the day, and then find his/her normal reporting time, after which a 0 or 1 is assigned to this factor that indicates a report is in normal time (0) or not (1). 3). For the day of week (*DOW*) factor, we divide a week into six parts, i.e., Monday to Friday, together with the Weekend, and then a number is used to label each report, e.g., 0 for Monday, and so on so forth. 4). And for the days to major version (*DTM*), we mainly mean the reporting days after a major version of the related project, and in this implementation, we roughly divide the time length by weeks, i.e., within one week after a major version, between one week and two weeks, and so on so forth, and reports that are over 8 weeks are gathered into one class, following the above style, we assign a number to each report with 0, 1 and so on.

The main contributions of this article are summarized as follows.

1. We propose a new feature, time factor, to examine the priority of the bug raised by the reports. Past researches have only considered factors on other dimensions but failed to figure out the importance of time factor in prioritization.
2. We examine how time factors affect the prioritization, and tell the extents of the optimization by adding the time factors to the model.
3. We have experimented our solution on huge numbers of bug reports to determine the priority of the bugs. The result shows that our solution can significantly improve the correctness and effectiveness of prioritization.

The rest of this paper is organized as follows. Section 2 presents the background and some related work, the intuitions of the proposed time factors is discussed in Section 3, and the comparing experiments are shown in Section 4, along with the discussion. Then finally, we conclude our work in Section 5.

2 Background and Related Work

Bug triage is needed in a bug repository to examine all the bugs reports entered into the repository. There are several features for the triagers to examine. Basically, whether the bug report is duplicate or not will be examined first. Then usually the validity of the bug report will be checked. The basic purpose of these judgments is to remove all the potential bug reports that are unnecessary to be resolved. After examining these features, triagers will take other features in the bug report to decide the severity and priority or change the former ones of the bug report in order to ensure that the most important bugs will be resolved quickly and well enough. Therefore, prioritization is meaningful in bug triage.

As we are going to prioritize the bug reports by judging their features, we decide to use two classification algorithms to build up our model in two different ways. The two algorithms are Support Vector Machine and Naive Bayes.

Support Vector Machines build non-linear models from training sets and are especially suitable for text classification. For instance, for two different categories in a plane, what SVM does is to draw a graph to separate these two categories as far as possible. The model it builds assigns new entities into one category or the others, which is a so-called non-probabilistic binary linear classifier. For prioritization, we have 5 classes to categorize (from P_1 to P_5), which means this is a multi-class classification. Naive Bayes, on the other hand, is just a simple probabilistic based on the Bayes' theorem. All the models are based on the hypothesis that every feature is independent from others. As the Bayes' theorem considers the probability of an event based different parameters or conditions, Naive Bayes simply uses this theorem to classify different categories according to several features given by the training set. Also, Naive Bayes has a good compatibility of classifying the textual information.

Since bug reports prioritization is put great emphasis on and studied a lot, what we have to do is to try to advance the precision of the prioritization. After reviewing the researches on prioritization, we find that most researchers neglect one of the factors that we are interested in. That is the time factor. Many researchers, however, take it unimportant and think that it is no use in predicting the prioritization. We, on the contrary, opposed to this idea and supposed that time factor can be critical to the prioritization, especially according to the influence of time on other factor. In the study of social network, time is an important feature to be considered. Since human activities are immensely influenced by the time, time should not be ignored. Similar to this thought, since bug reports are submitted by people who will be easily influenced by the factor of time, we think that the severity or the priority of the bug report is relevant to the time factor. Thus, we raised this research to find the relationship

between time factor and prioritization.

According to several researches made on bug triage, many attributes of the bug reports are used to determine the severity and priority of them. For example, Cubranic [5] has worked on recommending the bug reports according to their descriptions. By using the text categorization, they can decide which developer is responsible for a certain bug. Ahsan et al. [1] uses other features such as the titles and author of the bug reports in the classifier. These factors help the prediction become more accurate. Also, different measures for classification are taken to accomplish the goal. Kanwal et al. [7] used SVM and Naive Bayes to classify the bug reports and compare two different ways for their efficiency and precision. Anvik et al. [2, 3, 4] develops different classification models due to several machine learning techniques. After comparing each of the models, they gave out a brief overview of the advantages and disadvantages of using these models.

Also, there are many related works in studying the influence of time factor on other areas. For instance, Koutsonikola et al. [8] proposes a clustering framework which groups users according to their preferred topics and the time locality of their tagging activity. By this means, they can reveal the topic-domain of users interests and significantly contributes in a profile construction process. Sutton et al. [11] assigns credit by means of the difference between temporally successive predictions. They prove their convergence and optimality for special cases and relate them to supervised-learning methods. Wang et al. [13] presents an LDA-style topic model that captures not only the low-dimensional structure of data, but also how the structure changes over time, showing improved topics, better time-stamp prediction, and interpretable trends.

Some researches also do a lot in the field of machine learning, especially in SVM and Naive Bayes which we use in our research. Joachims et al. [6] did a lot in making large scale SVM learning practical. It presents algorithmic and computational results developed for SVMlight V2.0, which make large-scale SVM training more practical. The results give guidelines for the application of SVMs to large domains. McCallum et al. [10] simply empirically compared performance of two different ways of classification on five text corpora and shows the advantage and disadvantage of different Naive Bayes models, which gave us ideas about which way to choose for modeling the prioritization of the bug reports.

3 Intuitions

In this research, we divide the time factor of the bug reports into 4 different parts, i.e., *TOD*, *NT*, *DOW* and *DTM*. At first we chose these factors basically from our experience and intuition. Then we made several observations on these

factors and found some evidence about them.

Intuition 1: the severity or the priority is related to the time of the day.

This thought comes from our own experience, when we were suddenly called late in the night to solve an important bug. This bug report was handed in the mid-night, which is a rare time period for bug reports. Through this experience, we began to notice the time when a bug report is submitted. From our observation, those bug reports handed in a rare time period are more likely to be marked as severe and important. Thus, we believe that the severity or the priority is related to the time of the day

Intuition 2: the severity or the priority is related to the time whether a reporter submit a bug report in the time period he normally do.

Similar to our intuition 1, we suppose that a reporter would have different habits of reporting a bug, e.g., normally, he would report during the morning, when we notice a report that is reported in the afternoon, there is a high probability that this bug report has a high priority.

Intuition 3: the severity or the priority is related to the day of the week.

We suppose that the day in a week is also critical to the priority of a bug report. For instance, many developers may meet the case when they have to work overtime during the weekends, basically solving a bug reported on Friday or weekends. Some time especially when holidays and festivals will make the bug report rank higher and need to be solved at once. From this observation, we believe that severity and priority is related to the day of the week, and specifically, we divide a week by Monday to Friday, and the Weekend.

Intuition 4: the severity or the priority is affected by the days related to the latest major version.

When a new version of the software is released, it is quite common to see patches being installed soon after the release. Usually, the bug reports which are submitted most closest to the release date are thought the most important, mainly because these bugs are supposed to be the most obvious and easiest to be found. Thus, those bug reports submitted soon after the new major version released are always considered necessary to solve immediately.

4 Experiment

4.1 Data

In this paper, we collected our dataset from the Trac¹ Open Source Project. It is an enhanced wiki and issue tracking system for software development projects. It uses a minimalistic approach to web-based software project management. Mainly, we collected data of two projects, the

¹<https://trac.edgewall.org>

first is Trac itself, and the other is the Wordpress² project. And specifically, the collected bug reports for Wordpress was between *June, 2004* and *March, 2013*, while the bug reports for Trac project was between *August, 2003* and *July, 2013*. The statistics of these two datasets is shown in Table 1.

Table 1. Descriptive statistics of dataset

Project	# Bug reports	# Reporter	# Versions
Wordpress	23,848	6,013	18
Trac	10,416	4,701	11

From these reports, we extract several basic features, which include id, current status, title, type, reporter, owner, priority, milestone, component, severity, keywords, cc and description. Also, as we take time factor as the testing factors, other features that may be ignored by other models, such as open time, version, are also included in our features. To note, only those with status values such as ‘resolved’, ‘closed’, ‘confirmed’, ‘fixed’ or ‘duplicate’ will be used to train our model, others may be supplementary for our results. Figure 1 shows an example of the used bug report.

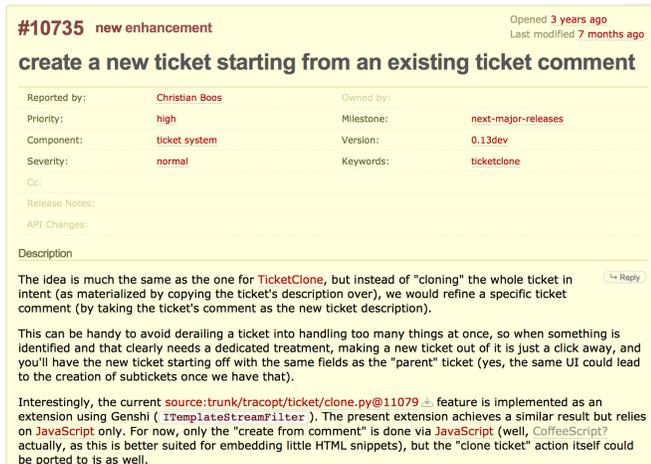


Figure 1. An example of Trac bug report

4.2 Metrics

We mainly use *Precision*, *Recall* to evaluate the effectiveness of our proposed approach.

Precision is defined as the ratio of relevant items that are predicted to all the predicted ones. In our experiments, relevant items are bug reports with priority that match the predicted priorities, so the *Precision* can be calculated by following equation:

²<https://wordpress.com>

$$Precision = \frac{Relevant \cap PredictedOnes}{NumberofPredicted}$$

Recall is the defined as the ratio of relevant items that are predicted to all the relevant ones. In our experiments, for one bug report, we have ground truth data of what priority of that bug report, so *Recall* can be defined as follows:

$$Recall = \frac{Relevant \cap PredictedOnes}{NumberofRelevant}$$

4.3 Significance Test

In order to find whether time factor have an influence on the prioritization of the bug reports, we performed the significance test to test the correlation between the 4 different kinds of time factors and the priority of the bug reports. Specifically, we adopt the Spearman correlation in our work, it is a non-parametric test that is used to measure the degree of association between two variables. Spearman correlation does not assume any assumptions about the distribution of the data and is the appropriate correlation analysis when the variables are measured on a scale that is at least ordinal. The formula used to calculate the Spearman correlation is as follows:

$$\rho = 1 - \frac{6 \sum d_i^2}{n(n^2 - 1)} \quad (1)$$

where ρ stands for the Spearman correlation, d_i is the difference between the ranks of corresponding values, and n is the number of values in each data set.

As a result, we got the following correlation for the four proposed time factors, as depicted in Table 2.

Table 2. The Spearman correlation of each time factor

Project	TOD	NT	DOW	DTM
Wordpress	0.022**	-0.014	0.006	-0.028**
Trac	-0.008	0.052**	-0.010	0.057**

** indicates the p-value < 0.01, which is significant.

From the result, we can see that time factor *DTM* is significant for both projects, and the *TOD* is effective for the Wordpress project, while the *NT* factor is effective for the Trac project. In our two chosen projects, the proposed time factor *DOW* is not significant according to the generated result.

4.4 Experimental results

Following the result of the significance testing, we evaluate to what extent these proposed time factors can affect

the task of bug report prioritization. First, we build up the model by using the basic features of the bug reports, which has been done by many researches before. Then, we build another model by incorporate the time factor as well as the basic features to see what are the differences. From the resulting data, we can know the influence of the time factor on the prioritization on each project.

For ease of understanding, in this experiment we only use the following basic features as to build the common classification models, i.e., the status, type, component, severity and reporter. And the classification models adopted in this paper are the SVM model and Naive Bayes.

In detail, a Support Vector Machine constructs a hyperplane or set of hyperplanes in a high or infinite dimensional space, then it can be used for classification. A good separation is achieved by the hyperplane that has the largest distance to the nearest training-data point of any class (so-called functional margin), since in general the larger the margin the lower the generalization error of the classifier. Naive Bayes is a simple technique for constructing classifiers, models that assign class labels to problem instances, represented as vectors of the feature values, where the class labels are drawn from some finite set. All different versions of Naive Bayes algorithms are based on a common principle: all naive Bayes classifiers assume that the value of a particular feature is independent of the value of any other feature, given the class variable.

For our case, there are mainly three sets of comparisons, i.e., 1). for the Wordpress project, we need to compare between basic features and after taking in the *TOD* time factor, 2). for the Trac project, we need will compare between the basic features based models and the with the *NT* time factor included, 3). and finally, for both projects, we should compare between the basic features based models and when the *DTM* time factor is considered. And for all these comparisons, the experiments are conducted on the above-mentioned two models.

Following we present the results of our comparisons.

Table 3 depicts the comparing result for the Wordpress project on the SVM model and Naive Bayes model, between only adopting the basic features and incorporating the *TOD* time factor. From which we can see that after taking the *TOD* time factor into consideration, the *Precision* and *Recall* are both improved by about 3%.

Table 3. *TOD* in Wordpress

Models	Project	Wordpress	
	Metrics	Precision	Recall
SVM	Basic	0.694	0.833
	With <i>TOD</i>	0.711	0.847
Naive Bayes	Basic	0.787	0.826
	With <i>TOD</i>	0.802	0.841

Table 4 illustrates the comparing result for the Trac project on the SVM model and Naive Bayes model, between simply adopting the basic features and incorporating the *NT* time factor. From which we can see by taking into account the *NT* time factor, the effectiveness of prioritization is evidently improved.

Table 4. *NT* in Trac

Models	Project	Trac	
	Metrics	Precision	Recall
SVM	Basic	0.571	0.756
	With <i>NT</i>	0.593	0.787
Naive Bayes	Basic	0.668	0.745
	With <i>NT</i>	0.689	0.782

Table 5 presents the comparing result between only using the basic features and adopting the *DTM* time factors, on the two models, for both of the two projects. From the result we can conclude that for both project, and using both SVM and Naive Bayes, after bringing in the *DTM* time factor, the ability of prioritization is improved.

Table 5. *DTM* in both projects

Models	Project	Wordpress		Trac	
	Metrics	Precision	Recall	Precision	Recall
SVM	Basic	0.694	0.833	0.571	0.756
	With <i>DTM</i>	0.726	0.853	0.597	0.799
NB	Basic	0.787	0.826	0.668	0.745
	With <i>DTM</i>	0.793	0.838	0.692	0.783

4.5 Threats to validity

In our research, all the threats to the validity come basically from the experimental errors. Though we checked the process and implementation of our experiments, there are still some errors that we could not avoid. As all the data we use are triaged and prioritized by human, many severity and priority may be subjective and vary from person to person, or even worse, sometimes they could be wrong. Relatively, time factor is objective and does not have too much influence on the validity. This is the threat to the internal validity. For external validity, the threat is that all the data we use come from one source. Though we used more than ten thousand reports, we track all these bug reports from the Trac Open Source Project, which means that we could not be comprehensive to all the sources of bug reports. In the future, if time permits, we would probably track more bug reports from other sources, which can increase the reliability and sustainability of our research.

5 Conclusion

This paper introduces the time factor in the bug report prioritization. Specifically, we investigated four time factors, i.e., the time of the day, the normal time, day of week and days to the latest major version. For our specific case, the *TOD* time factor and *DTM* time factor is effective for project Wordpress, while for the Trac project the *NT* time factor and *DTM* time factor is effective. Our experimental results demonstrate that incorporating the time factors can effectively improve the *Precision* and *Recall* for bug report prioritization. In future, we will consider evaluating our proposal on more sources of data, as well as adopting more sorts of classification models.

References

- [1] S. N. Ahsan, J. Ferzund, and F. Wotawa. Automatic software bug triage system (bts) based on latent semantic indexing and support vector machine. In *Software Engineering Advances, 2009. ICSEA'09. Fourth International Conference on*, pages 216–221. IEEE, 2009.
- [2] J. Anvik. Automating bug report assignment. In *Proceedings of the 28th international conference on Software engineering*, pages 937–940. ACM, 2006.
- [3] J. Anvik, L. Hiew, and G. C. Murphy. Coping with an open bug repository. In *Proceedings of the 2005 OOPSLA workshop on Eclipse technology eXchange*, pages 35–39. ACM, 2005.
- [4] J. Anvik, L. Hiew, and G. C. Murphy. Who should fix this bug? In *Proceedings of the 28th international conference on Software engineering*, pages 361–370. ACM, 2006.
- [5] D. Čubranić. Automatic bug triage using text categorization. In *In SEKE 2004: Proceedings of the Sixteenth International Conference on Software Engineering & Knowledge Engineering*. Citeseer, 2004.
- [6] T. Joachims. Making large scale svm learning practical. Technical report, Universität Dortmund, 1999.
- [7] J. Kanwal and O. Maqbool. Bug prioritization to facilitate bug report triage. *Journal of Computer Science and Technology*, 27(2):397–412, 2012.
- [8] V. Koutsonikola, A. Vakali, E. Giannakidou, and I. Kompatsiaris. *Clustering of social tagging system users: A topic and time based approach*. Springer, 2009.
- [9] T. Kremenek and D. Engler. Z-ranking: Using statistical analysis to counter the impact of static analysis approximations. In *Static Analysis*, pages 295–315. Springer, 2003.
- [10] A. McCallum, K. Nigam, et al. A comparison of event models for naive bayes text classification. In *AAAI-98 workshop on learning for text categorization*, volume 752, pages 41–48. Citeseer, 1998.
- [11] R. S. Sutton. Learning to predict by the methods of temporal differences. *Machine learning*, 3(1):9–44, 1988.
- [12] Y. Tian, D. Lo, and C. Sun. Drone: Predicting priority of reported bugs by multi-factor analysis. In *2013 IEEE International Conference on Software Maintenance*, pages 200–209. IEEE, 2013.
- [13] X. Wang and A. McCallum. Topics over time: a non-markov continuous-time model of topical trends. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 424–433. ACM, 2006.

A Model for Predicting Bug Fixes in Open Source Operating Systems: an Empirical Study

Paolo Ciancarini
Università di Bologna and CINI
Italy
paolo.ciancarini@unibo.it

Alberto Sillitti
Innopolis University and CINI
Russian Federation
a.sillitti@innopolis.ru

Abstract—This paper proposes an adaptation to the open source environment (Linux Kernel and OpenSolaris) of a model for predicting which bugs get fixed in the Microsoft Windows operating system. We have analyzed the entire bug repositories containing 16,136 bug reports reported in about 8 years of activity of the project (from 2002 to 2010) for the Linux Kernel and 16,301 bug reports reported in about 3 years of activity of the project (from 2007 to 2010) for OpenSolaris. According to the data analyzed and the descriptive models produced, we have found that (a) bugs reported by people with better reputation and bugs in which more people are involved are more likely to get fixed, (b) reassigning or re-opening bugs are not affecting the fix likelihood, and (c) managing bugs in the same location increases the fix likelihood. The predictive model defined has a precision of 61% and a recall of 39% for the Linux Kernel and a precision of 76% and a recall of 73% for OpenSolaris. These results are comparable with the ones for Microsoft Windows. document.

Keywords—bugs; predictive models; open source

I. INTRODUCTION

Bugs are one of the major problems in software development since the very early era of computers (Brooks, 1995). In any kind of software products, not all bugs are fixed. This is because different bugs have a different impact on the user and bug fixes requires a significant amount of time and effort (Ciancarini et al., 2015; Ciancarini et al., 2016; Di Bella et al., 2013b; Pedrycz et al., 2015; Remencius et al., 2016). Moreover, in popular products, the number of bug reported is usually very high and there are not enough resources to fix them all. As discussed in Guo et al. (2010), a significant amount of effort is devoted to bugs that remain unfixed: all this effort is wasted. For these reasons, it is essential to identify as fast as possible the bugs that are fixed and the one that remain not fixed. To support this identification many different approaches are possible based on the analysis of the development process (Abrahamsson et al., 2007; Di Bella et al., 2013; Petrinja et al., 2010; Sillitti et al., 2012) and the analysis of the code (Jermakovics et al., 2008; Jermakovics et al., 2011; Pedrycz et al., 2002).

This paper adapts the work of Guo et al. (2010) developed on the Microsoft Windows operating system to open source alternatives, namely the Linux Kernel and OpenSolaris. To adapt better the model, we have considered the development of the two systems till 2010 since the original study was of that

year. In this study, we have investigated the Linux Kernel that was about 12 MLOC and OpenSolaris, about 20 MLOC.

To this end, the major outcomes of the work of Guo et al. (2010) and ours are summarized in Table I.

TABLE I. MAJOR FINDINGS OF THE PAPER OF GUO ET AL. (2010) COMPARED TO OUR STUDY

Guo et al. (2010)	Our study
“People more successful in getting their bug fixed in the past are more likely to get their bugs fixed in the future.”	Confirms
“The more people are involved in the bug life cycle, the more likely it is fixed.”	Confirms
“Reassignments of bugs are not always detrimental to the likelihood of bug fix.”	Confirms
“Re-openings of bugs are not always detrimental to the likelihood of bug fix.”	Confirms
“Bugs assigned across teams or locations are less likely to get fixed.”	Confirms
Definition of a model for predicting the probability of fix of a given bug at the time of opening.	Similar model

This paper is structured as follows: Section II presents the state of the art of the area; Section III describes the research methodology and the data; Section IV and V provide a descriptive and a predictive statistical models; finally, Section VI draws the conclusions and provides directions of future work.

II. RELATED WORK

There are several studies related to bug reports. In particular, there are several papers related to bug analysis and prediction. However, in our knowledge, the only study related to the likelihood of fix of a bug is the one by Guo et al. (2010) related to the bug reports of the Microsoft Windows operating system. Moreover, no studies are available in the open source area.

Readers interested in an overview of the state of the art and of the related literature can refer to the paper of Guo et al. (2010).

III. RESEARCH METHODOLOGY

A. Environment

We have performed a quantitative analysis extending the approach followed by Guo et al. (2010) to the analysis of open source projects. In particular, we have considered the bug

repositories of the Linux Kernel and OpenSolaris till 2010. We have selected such software because they are both open source operating systems and the availability of a large amount of information. In particular, all the bug reports are publicly available. Moreover, the timeframe is limited to 2010 since we compare with the study of that year.

The main differences with the study of Guo *et al.* (2010) are related to the open nature of the considered projects, in particular: a) we had no possibility to propose a questionnaire to the developers; b) we had no information about the actual geographical location of the developers; c) we had no information about the organizational structure of the contributors, except from the information we can get from their email addresses that is basically the organization they belong to. In particular, we were not aware if people in the same organization share the same boss and/or belong to the same working team.

Even with the listed differences, it was possible to compare the core part of the studies and propose an approach to make a similar analysis possible in any public bug repository.

B. Data

We have considered all the bugs stored in the bug tracking systems of the Linux Kernel and the OpenSolaris operating systems till 2010.

For each bug report, we extracted the history of the bug and the following set of information:

- **Editor:** who has modified the bug report
- **State:** if the bug is still open or not
- **Component:** the name of the affected component
- **Severity:** potential impact of the bug on the system: blocking, major, minor, trivial, etc.
- **Opener:** who has opened the bug
- **Assignee:** who is assigned to manage the bug
- **Resolution:** if the bug has been resolved. The study considers only bugs that are marked as resolved as fixed or not. Such other status includes bugs that are identified as duplicates, bugs that will not be fixed, etc.

Compared to the study of Guo *et al.* (2010), we were unable to collect the following information:

- **Bug source:** the source of the bug report such as internal testing, a customer, code review, etc.
- **Bug type:** bug in code, specification, test, etc.

In the projects considered, the typical life cycle of a bug is the following: when a bug is opened, all the described fields are filled in by the opener, than the bug is edited one or more times (including reassignments to other developers) until the bug is resolved. However, it might happen that a resolved bug is reopened because it was not solved properly.

Our goal is to characterize bugs that have been resolved successfully and in the case of re-openings, we consider only the final status in 2010.

Beside the bug reports, Guo *et al.* (2010) presented data related to a questionnaire that was filled in by a large number of developers and data related to the organization of the development teams in Microsoft. In our study, these data were not available since our data come from open source projects. The data we miss are the following:

- **Geographical data:** information about the co-location of people in the same office, building, or campus.
- **Organizational data:** information developers belonging to the same team.
- **Qualitative data:** information collected through a questionnaire asking the developers about the factors that influence the likelihood for a bug of being fixed.

According to their results, geographical and organizational data have some impact on the likelihood for a bug of being fixed. However, these factors have a limited impact in open source projects since most of them are extremely distributed and do not have a hierarchical organization.

IV. DESCRIPTIVE STATISTICAL MODELS

A. Building the models

We have built two models to predict the probability that a bug will be fixed in the two operating systems considered. To build the models, we have used the same approach described in Guo *et al.* (2010). The identified factors and the related coefficients are listed in Table II. We also found that all the significant factors are statistically significant at $p < 0.001$, according to an Analysis of Deviance chi-square test (Hosmer and Lemeshow, 2000). We have checked for interactions between factors and we have verified that there are no statistically significant interactions.

TABLE II. DESCRIPTIVE LOGISTIC REGRESSION MODEL FOR BUG FIX PROBABILITY. "N.A." VALUES REFER TO FACTORS THAT ARE NOT AVAILABLE IN THE SPECIFIC MODEL; "N.S." VALUES REFER TO FACTORS THAT ARE NOT STATISTICALLY SIGNIFICANT IN THE SPECIFIC MODEL.

Factor	Coefficient		
	Windows Vista	Linux Kernel	OpenSolaris
Reputation of bug opener	2.19	2.03	1.83
Reputation of 1 st assignee	2.46	0.26	2.39
Opened by a temporary employee	-0.12	n.a.	n.a.
Initial severity level	0.03	n.s.	0.06
Opener/any assignee same manager	0.68	n.a.	n.a.
Opener/any assignee same building	0.27	n.a.	n.a.
Severity upgrade	0.26	n.s.	0.28

Factor	Coefficient		
	Windows Vista	Linux Kernel	OpenSolaris
Editors	0.24	n.s.	0.11
Assignee buildings	-0.26	n.a.	n.a.
Re-openings	-0.13	n.s.	n.s.
Component changes	-0.23	-0.20	-0.20
Opener and 1 st assignee in the same organization	n.a.	0.47	n.s.
Opener/any assignee same organization	n.a.	n.s.	0.47
Opener and 1 st assignee are the same person	n.a.	n.s.	1.04
Assignees	n.a.	0.18	n.s.
Comments	n.a.	0.01	n.s.
Severity changes	n.a.	n.s.	-0.31

The purpose of this model is to describe the factors affecting the bug fixes but it cannot be used to make predictions since it includes factors that are not available at the time of creation of the bug report. A predictive model is described in Section V.

B. Meaning of the regression coefficients

As it happens in Guo *et al.* (2010), the meaning of the parameters of the logistic regression model is intuitive. The list of factors presented includes all the factors coming from a quantitative experimentation (in the case of Windows Vista, the study of Guo *et al.* (2010) included some factors collected through questionnaires, therefore these have been omitted).

Moreover, some of the factors considered in Guo *et al.* (2010) are not considered in our models:

- **Opened by a temporary employee:** this factor does not have any meaning in open source projects since a significant amount of contributions is provided by volunteers (the percentage of paid developers and volunteers varies a lot in different open source projects)
- **Opener/any assignee same manager:** this factor is not considered in our models for the same reason as the previous factor.
- **Opener/any assignee same building:** this factor is not considered in our models since we are not aware of the physical location of the developers. In many open source projects most of the developers are geographically distributed.

The last 6 factors are not included in Guo *et al.* (2010) for different reasons:

- **Opener and 1st/any assignee in the same organization:** these were obviously not considered since all the development was inside Microsoft.
- **Opener and 1st assignee are the same person:** this factor is analyzed in the paper (even if no values are

reported for confidentiality) but it is not present in the proposed model.

- **Assignees:** the authors in Guo *et al.*, 2010 preferred to use the number of assignee buildings. However, in our models we are unable to do it since we do not know the location of each developer.
- **Comments:** this factor is not considered in the original paper. In our models it has a very small effect and it is statistically significant only for the Linux Kernel.
- **Severity changes:** this factor is not considered in the original paper that considers only the fact that the severity has been upgraded, downgraded, or not modified. This factor has a relevant effect in the OpenSolaris model while it is not significant in the Linux Kernel one.

C. Interpretation of the models

According to the two developed models, the following factors are correlated with the bug fix probability:

- **Reputation of bug opener and of the 1st assignee:** as discussed in Section IV.C, these factors are strongly positively correlated with the likelihood of fix in both the analyzed projects (actually the correlation is weaker for the 1st assignee in the Linux Kernel but it is still relevant). This could mean that bug openers with a relevant track record in having fixes are going to receive more fixes in the future. This could be because of the effectiveness of their reports. The same is true for the 1st assignee since he could be effective in fixing the bug (effective management of the assignment of bugs to the correct person) or he can redirect the bug to the best person able to fix it. In any case, it is linked with an effective management of the assignment of the bugs.
- **Initial severity level:** this factor has a positive correlation with the likelihood of fix of a bug. However, this is statistically significant only in the OpenSolaris model. This could be related to the fact that the number of severity changes and severity upgrades are not significant in the Linux Kernel model. Therefore, the significance of this factor could also be related to the same motivation: a more structured management of the severity levels that might occur in OpenSolaris compared to the Linux Kernel. This is also supported by the fact that Guo *et al.* (2010) in Microsoft reports a similar value.
- **Severity upgrade and Severity changes:** changes in severity affects the likelihood of bug fix. However, in our two data sets, they affect only OpenSolaris, while they are not significant in the Linux Kernel. This behavior could be linked with a more structured management of the severity levels that might occur in OpenSolaris compared to the Linux Kernel since a

consistent part of its development was done inside Sun Microsystems (now Oracle) (therefore, the corporate organization could be more similar to the Microsoft one reported in Guo *et al.* (2010)).

- **Editors:** the number of editors are positively correlated with the likelihood of fix, however this factor is significant only for the OpenSolaris model. This could also be related to the more structured contributions from Sun Microsystems, as editors and edits could be linked more strictly with the knowledge of the system and a deeper investigation of the bug (and maybe more information in the bug reports).
- **Component changes:** this factor is negatively correlated with the likelihood of bug fix. This happens in both models and could be related to a vague bug report that is not able to identify precisely the source of the error.
- **Opener and 1st assignee in the same organization and Opener/any assignee same organization:** these factors are positively correlated with the likelihood of bug fix, however only one of them is statistically significant in each of the two models. In any case, they identify a link between the opener and the management of the bug inside the same organization. This could relate the interest/usage of a specific set of functionalities of a product and the involvement in its development.
- **Opener and 1st assignee are the same person:** this factor is positively correlated with the likelihood of bug fix. However, it is statistically significant only in the OpenSolaris model. This could be related to the size of the community since Linux is more adopted than OpenSolaris, therefore the factor is not affecting significantly the likelihood of bug fix. On the contrary, in OpenSolaris the community of reporters maybe overlap more with the community of developers.
- **Assignees:** this factor is positively correlated with the likelihood of fix, however this factor is significant only for the Linux Kernel model. In any case, this factor is linked to the number of people involved in the fix of a bug. This could be related to the community approach to software development in open source projects (even if OpenSolaris is open source, its history as an open product is quite recent and a large part of its community was linked to Sun Microsystems).
- **Comments:** this factor has a positive correlation with the likelihood of fix of a bug. However, this is statistically significant only in the Linux Kernel model. This could be related to the community approach to software development in open source projects as described in the previous factor.

D. Comparison with other studies

In our knowledge, the only comparable study is the one of Guo *et al.* (2010). The main similarities are the following:

- The reputation of bug opener and of the 1st assignee are very important to determine if a bug will be fixed. For the bug opener our two models present coefficients close to the value of the study of Guo *et al.* (2010), for the 1st assignee the strongest similarity is between the original study and the OpenSolaris model. This could happen because even if OpenSolaris is open source, its roots are in the Solaris operating system developed at Sun Microsystems that provides a large amount of contributions to the project.
- Re-opening or reassigning a bug does not affect the bug fix likelihood.
- Bugs assigned across teams (organizations in our data sets) are less likely to get fixed.
- Changing the component to which the bug is associated decreases the bug fix likelihood. It is an indication that it is not clear where the problem is and/or the bug report is not adequate to identify the problem.

And the main differences are:

- We are unable to evaluate the role of the hierarchy of the organization in the bug fix likelihood. This is because in our projects the interactions among developers are more complex than in the study of Guo *et al.* (2010). In open source projects there is a mixture of contributions including different companies, associations, and volunteers. However, if the opener and assignees belongs to the same organization (or even it is the same person) the bug fix likelihood increases.
- The number of comments in a bug report increases the bug fix likelihood in the Linux Kernel model. This could be because of the collaborative approach of the development in open source projects (OpenSolaris does not have a long tradition of community-based development).
- Severity upgrades and changes affect positively and negatively only the fix likelihood of OpenSolaris, this could be related to a more structured management of such information as it happens in Microsoft.

Moreover, the two studies differ also about some collected data due to the different environments considered.

Some of the factors considered in the study of Guo *et al.* (2010) are not included in our model:

- **Opened by a temporary employee:** this factor does not have any meaning in open source projects since a significant amount of contributions is provided by volunteers (the percentage of paid developers and

volunteers varies a lot in different open source projects)

- **Opener/any assignee same manager:** this factor is not considered in our model for the same reason as the previous factor.
- **Opener/any assignee same building:** this factor is not considered in our model since we are not aware of the physical location of the developers. In many open source projects most of the developers are geographically distributed.

However, we have extended the model proposed by Guo *et al.* (2010) with 6 factors that are relevant for open source projects:

- **Opener and 1st/any assignee in the same organization:** these are obviously important in open source development since the development is performed in a distributed way with contributions of several organizations.
- **Opener and 1st assignee are the same person:** this factor has the same motivation as the one before.
- **Assignees:** our models use such information while the paper of Guo *et al.* (2010) preferred to use the number of assignee buildings. Considering open source development, it is usually difficult to know the location of each developer.
- **Comments:** comments are very useful in open source development since the communication among developers is usually mediated by tools. However, in our models, it has a very small effect and it is statistically significant only for the Linux Kernel.
- **Severity changes:** this factor could be an indicator of the policies for managing bugs in the project. It has a relevant effect in the OpenSolaris model, while it is not significant in the Linux Kernel one.

The model we have proposed is easily replicable in any open source project since it is based on information that nearly all open source projects provide.

V. PREDICTIVE STATISTICAL MODELS

A. Building the models

Using the experience in the development of the descriptive model described in the Section IV, we have built two models to predict the probability that a bug will be fixed in the two operating systems considered. To do that, we have replicated the approach of Guo *et al.* (2010). The identified factors and the related coefficients are listed in Table III. Also in this case, we found that all the factors are statistically significant (mostly at $p < 0.001$) according to an Analysis of Deviance chi-square test (Hosmer and Lemeshow, 2000). We have checked for interactions between factors and we have verified that there are no statistically significant interactions.

In particular, the predictive models include only factors that are known at the time of the bug submission, therefore only a subset of the factors of the descriptive models are included in the new one. Obviously, the related coefficients are slightly different.

TABLE III. PREDICTIVE LOGISTIC REGRESSION MODEL FOR BUG FIX PROBABILITY. "N.A." VALUES REFER TO FACTORS THAT ARE NOT AVAILABLE IN THE SPECIFIC MODEL; "N.S." VALUES REFER TO FACTORS THAT ARE NOT STATISTICALLY SIGNIFICANT IN THE SPECIFIC MODEL; "*" IS SIGNIFICANT AT $P < 0.05$; "**" IS SIGNIFICANT AT $P < 0.01$.

Factor	Coefficient		
	Windows Vista	Linux Kernel	OpenSolaris
Reputation of bug opener	2.19	1.75	1.74
Reputation of 1 st assignee	2.39	0.18*	2.63
Opened by a temporary employee	-0.04	n.a.	n.a.
Initial severity level	0.06	n.s.	0.04**
Opener/any assignee same manager	0.27	n.a.	n.a.
Opener/any assignee same building	0.03	n.a.	n.a.
Opener and 1 st assignee in the same organization	n.a.	0.48	0.53
Opener and 1 st assignee are the same person	n.a.	n.s.	0.80

B. Performances of the models

We have investigated the precision and the recall of the developed models and even if our models are based on less information than the ones of Guo *et al.* (2010), we have obtained comparable performances. In particular, we have obtained a precision of 0.61 and a recall of 0.41 for the Linux Kernel model and a precision of 0.76 and a recall of 0.73 for the OpenSolaris one.

Such results have been obtained applying the same approach described in Section IV.

C. Comparison with other studies

Since in our knowledge, the only similar study is the one of Guo *et al.* (2010), we have compared our model with theirs (Table IV). In particular, the Linux Kernel model performs worse than the original model, while the OpenSolaris one performs better. This could be related to the fact that even if OpenSolaris is an open source product, it comes from a closed source system and still mostly developed by Sun Microsystems.

TABLE IV. COMPARISON OF THE PRECISION AND RECALL OF THE MODELS

	Guo <i>et al.</i> , 2010		Our study	
	Windows Vista	Windows 7	Linux Kernel	OpenSolaris
Precision	0.67	0.68	0.61	0.76
Recall	0.68	0.64	0.41	0.73

VI. CONCLUSIONS AND FUTURE WORK

This study aimed at extending in an open source environment an empirical investigation performed inside Microsoft. Our results has confirmed the claims of the study of Guo *et al.* (2010) and was able to build two predictive models with comparable performances using less information.

We think that our study provides a contribution to the generalizability of the findings of empirical studies in software engineering and we aim at replicating the study in several more open source projects.

ACKNOWLEDGMENT

This paper has been partially supported by Consorzio Interuniversitario Nazionale per l'Informatica (CINI) through the ECSEL project MANTIS (662189).

REFERENCES

- [1] Abrahamsson P., Moser R., Pedrycz W., Sillitti A., Succi G., "Effort Prediction in Iterative Software Development Processes – Incremental Versus Global Prediction Models", 1st International Symposium on Empirical Software Engineering and Measurement (ESEM 2007), Madrid, Spain, 20 - 21 September 2007.
- [2] Brooks F. P., *The Mythical Man-Month: Essays on Software Engineering*, Addison-Wesley, 2nd edition, 1995.
- [3] Ciancarini P., Poggi F., Rossi D., Sillitti A., "Improving bug predictions in multi-core cyber-physical systems", 4th International Conference on Software Engineering for Defense Applications (SEDA 2015), Rome, Italy, 26 - 27 May 2015.
- [4] Ciancarini P., Poggi F., Rossi D., Sillitti A., "Mining Concurrency Bugs", Embedded Multi-Core Systems for Mixed Criticality Summit 2016 at CPS Week 2016, Vienna, Austria, 11 April 2016.
- [5] Di Bella E., Sillitti A., Succi G., "A multivariate classification of open source developers", *Information Sciences*, Elsevier, Vol. 221, pp. 72 - 83, February 2013.
- [6] Di Bella E., Fronza I., Phaphoom N., Sillitti A., Succi G., Vlasenko J., "Pair Programming and Software Defects – a large, industrial case study", *Transaction on Software Engineering*, IEEE, Vol. 39, No. 7, pp. 930 - 953, July 2013.
- [7] Guo P. J., Zimmermann T., Nagappan N., Murphy B., "Characterizing and Predicting Which Bugs Get Fixed: An Empirical Study of Microsoft Windows", *32nd International Conference on Software Engineering (ICSE 2010)*, Cape Town, South Africa, 2 - 8 May, 2010.
- [8] Hooimeijer P., Weimer W., "Modeling bug report quality", *22nd IEEE/ACM International Conference on Automated Software Engineering (ASE 2007)*, Atlanta, GA, USA, 5 - 9 November, 2007.
- [9] Hosmer D. W., Lemeshow S., *Applied Logistic Regression*, John Wiley & Sons, 2nd edition, 2000.
- [10] Jermakovics A., Moser R., Sillitti A., Succi G., "Visualizing Software Evolution with Lagrein", 22nd Object-Oriented Programming, Systems, Languages & Applications (OOPSLA 2008), Nashville, TN, USA, 19 - 23 October 2008.
- [11] Jermakovics A., Sillitti A., Succi G., "Mining and Visualizing Developer Networks from Version Control Systems", 4th International Workshop on Cooperative and Human Aspects of Software Engineering (CHASE 2011) at ICSE 2011, Honolulu, HI, USA, 21 May 2011.
- [12] Pedrycz W., Succi G., Chun M. G., "Association Analysis of Software Measures", *International Journal of Software Engineering and Knowledge Engineering*, Vol. 12, No. 3, pp. 291 - 316, 2002.
- [13] Pedrycz W., Succi G., Sillitti A., Iljazi J., "Data description: A general framework of information granules", *Knowledge-Based Systems*, Elsevier, Vol. 80, pp. 98 - 108, May 2015.
- [14] Petrinja E., Sillitti A., Succi G., "Comparing OpenBRR, QSOS, and OMM Assessment Models", 6th International Conference on Open Source Systems (OSS 2010), Notre Dame, IN, USA, 30 May - 2 June 2010.
- [15] Remencius T., Sillitti A., Succi G., "Assessment of software developed by a third-party: A case study and comparison", *Information Sciences*, Elsevier, Vol. 328, pp. 237 - 249, January 2016.
- [16] Sillitti A., Succi G., Vlasenko J., "Understanding the Impact of Pair Programming on Developers Attention: A Case Study on a Large Industrial Experimentation", 34th International Conference on Software Engineering (ICSE 2012), Zurich, Switzerland, 2 - 9 June 2012.

Exploring DevOps for Data Analytical System with Essential Demands Elicitation

Jiabin ZHENG*, Yan LIU†

School of Software Engineering, Tongji University

Shanghai, China

Email: *1434321@tongji.edu.cn, †yanliu.sse@tongji.edu.cn

Jin Lin‡

Shanghai Maitrox Electronics Co., Ltd

Shanghai, China

Email: ‡jeffrey@maitrox.com

Abstract—DevOps is an emerging concept and methodology for bridging the gap in the process of software development. At present, applying DevOps to data analytical system (DAS) is increasingly embraced. But the characteristics of this system, such as data protection, always leads to a series of constrains. It's a bit difficult to conduct DevOps on data analytical system. Moreover, there are no DevOps solutions for reference. Therefore, exploring DevOps for data analytical system is valuable.

In this paper, we illustrate DevOps demands of data analytical system from different perspectives, and constantly emphasize the importance of automation toolchain. Based on them, a process model for DAS DevOps (*D²Ops*) is proposed to clarify participants activities. In order to improve the efficiency, we attempt to integrate the automation toolchain. With the consideration of stability, six generic process components are designed to support this model. They can be the selection criteria for specific automation tools. We also present a reference facility based on these generic process components, and illustrate its implementation combing with a practical case.

Keywords—DevOps; Data Analytical System; Constraint; Automation Tool; Process Component;

I. INTRODUCTION

Along with the vigorous development of Internet, large amounts of data are produced daily in various fields. For extracting valuable informations from raw data, the data analytical system has become a rising trend nowadays. However, it probably has a more complex process and more different participants compare with general system. To reduce the difficulty of system development, DevOps, an emerging concept and methodology, is brought in to optimize process.

DevOps originally derives from the demands of solving the misunderstanding and collision between development and operations. Now it involves with not only development and operations, but also integration, delivery, quality assurance and other relevant aspects[1]. DevOps describes practices that streamline the software delivery process, emphasizing the learning by streaming feedback from production to development and improving the cycle time[2]. At present, numerous open-source automation tools have been supported this rising concept. A few practical successful cases also demonstrate its significance. As a result, effective DevOps can be regarded as a promising way for data analytical system.

But there are inevitable challenges. They will be a great problem for carrying out DevOps.

- *No solutions for reference* - In essence, DevOps is only a conceptual principle. It doesn't provide any practice instruction. Consequently, the lack of standardization results in no common solutions. Almost all of the existing DevOps solutions are conducted empirically according to technology actuality. It's hard to take them for reference.
- *Constraints of system characteristic* - Data analytical system is quite different from other systems. It should put great efforts on data protection. Moreover, the result of data analysis is influenced by various factors. These system characteristics lead to constraints that may break the normal DevOps process.
- *Complexity of tool integration* - Massive open-source automation tools will be applied to DevOps practice. But there are too many various tools in the open source community. How to pick appropriate tools and how to integrate them into a full toolchain are bothering people.

To overcome obstacles brought by these challenges, our research focuses on exploring DevOps for data analytical system. The contribution includes:

- Analyze DevOps demands for data analytical system
- Propose a reference process model to clarify DevOps participants and their collaboration
- Design six generic components to support the envisaged DevOps process model

The remainder of this paper is structured as follows. In Section II we conclude the related achievements about DevOps in both of academia and industry. The fundamentals of our research are shown in Section III. We analyze the *D²Ops* demands of different perspectives. With the consideration of DevOps participants, Section IV and Section V propose a reference process model. Meanwhile, six generic process components are designed to support this model. Based on them, we demonstrate a reference facility and analyze its essential workflows. In Section VI, we present the facility implementation combined with a practical case. Finally, in Section VII, we illustrate the shortages of our research and make a plan for future work.

II. RELATED WORK

A. Academic Research

Walls[3] encouraged to build DevOps culture, and illustrated the key aspects of promoting success. By applying a qualitative analysis approach, Bang et al.[4] investigated the supporting way of factors, like knowledge, skills and abilities, for DevOps. Lwakatare[5] identified four main dimensions of DevOps: collaboration, automation, measurement and monitoring. Moreover, he developed a conceptual framework to facilitate the understanding of DevOps phenomenon.

It's apparent that all the mentioned above belong to conceptual researches that emphasize on culture propagation and conduction principles. But except those conceptual researches, achievements related to practice shouldn't be neglected. For example, Stillwell et al.[1] summarized their DevOps approach in HARNESS, a EU research project. Based on DevOps, Ferry et al.[6] proposed a continuous deployment solution for multi-cloud systems.

While, by investigation, vast majority of academic achievements are conceptual until now. As the lack of standard practice solutions and metrics for reference, DevOps practice is usually in a mess and hardly estimates whether it is reasonable.

B. Industrial Exploration

Now organizations in industry also constantly propose DevOps solutions for benefiting from them[7]. For instance, PayPal proposed a solution for continuous integration in the Hybrid Cloud. Cisco designed a solution in a SDN world. IBM implemented a generic solution of continuous delivery. While most of them are based on organization's technology actuality regardless of universality. No two enterprise-adopted DevOps approaches would be similar as each enterprise has unique characteristics and requirements[8]. Thus it's a bit difficult to apply these industrial DevOps solutions for us. Now this problem has been gradually realized. Some corporations like Microsoft, IBM and Amazon, attempt to establish common integrative DevOps solutions[9][10][11]. But these solutions are mostly based on their own cloud platforms that are inappropriate for confidential data.

C. Innovative Achievement

Apart from previous researches, now researchers have innovatively tried to combine DevOps with other fields. It's no longer confined to software development process. Babar et al.[8] utilized the technology of Business Process Architecture modeling to select customized DevOps processes. Bruneo et al.[12] used DevOps concept to develop an adaptive cloud management platform. Kim et al.[13] even discussed the DevOps requirements specific to the modern network service creations, and proposed an extended DevOps concept.

III. DEMAND ANALYSIS FOR D²OPS

A. Functional Demand

As similar with general DevOps solutions, the key points of functional demand for *D²Ops* include: (a) cost-saving; (b) automation; (c) the integrated toolchain; (d) data governance.

According to the statistics[14][15], DevOps has a remarkable effect on slashing cost. To a large degree, the effect is established on automation mechanism. A major advantage of automation is that tasks can be executed and information can be collected relatively more accurately and quickly as compared to manual approaches[16].

The implementation of automation requires a strong supporting from tools[17]. But processes are more important than tools for DevOps[2]. If applied in complete isolation, tools couldn't play their roles effectively. Then it's rather beneficial to combine processes with tools. Generally speaking, toolchain is a collection that organizes tools in accordance with a specific process. By contrast with the way of gathering tools simply, the integrated toolchain is more able to contribute to immense automation effects.

In addition, data quality needs major concerns. It's usually awful because data often come from various sources. The bad data quality may lead to a failure of constructing data models, and even disable the development process. Therefore, data governance technology is needed to ensure good data quality.

B. Premised Constraints

There are several special characteristics in data analytical system. On one hand, people are obliged to put enormous energy on data protection. On the other hand, the result of data analysis is influenced by many factors. These characteristics are the premises of data analytical system. They will result in some constraints.

- *Network Accessing Control* - Network is one of the most insecure tools now. Hence, network accessing control must be reinforced. For this purpose, a simple solution is the setup of LAN (Local Area Network) environment. This solution is often applied as a basic protection for data center. It aims to cut all channels to the Internet.
- *Data Accessing Restriction* - In order to limit the behavior of arbitrary data accessing, production environment and development environment should be separated. This isolation strategy is able to prevent the real data from being accessed outside. Furthermore, developers can still utilize the Internet resources.
- *Data Information Protection* - In the process of development, it's usually unavoidable to take advantage of third-party platforms. However, data-related informations, like source code, are able to reflect real data contents. Once informations are uploaded to those platforms, it may lead to confidential data leakage indirectly. Thus data informations should be strictly controlled.
- *Traceability of Analysis Result Quality* - Data analytical system depends heavily on visualization to represent data analysis result. An unexpected result is probably caused by various factors, like wrong visualization implementation, wrong data model implementation or wrong data model. It's hard to trace the root cause of quality issues.

As TABLE I shows, these constraints will bring about some latent issues. Although common solutions are able to cope with

TABLE I. Constraints Analysis for D^2Ops

Constraint	Latent Issues	Common Solutions	DevOps Enhancement
Network Accessing Control	<ol style="list-style-type: none"> 1. Cannot depend on web-based services in production system; 2. Cannot utilize public repositories on the Internet for server management; 	<ol style="list-style-type: none"> 1. Abandon code packages that need to connect with Internet; 2. Manually install tools and configure without Internet; 	<ol style="list-style-type: none"> 1. Set up local code package repositories like Nexus; 2. Build local tool repositories like local yum and local apt-get;
Data Accessing Restriction	<ol style="list-style-type: none"> 1. Cannot construct and verify data models in development environment; 2. Disable continuous deployment process; 	<ol style="list-style-type: none"> 1. Generate test data based on real data; 2. Manually deploy by transmission medium; 	Utilize virtualization technology for deployment
Data Information Protection	Cannot upload data-related informations to third-party platforms like GitHub	Refuse to use data-related third-party platforms	Host function-similar platforms in local server
Traceability of Analysis Result Quality	Hard to confirm what causes the unexpected result and take too much time	Implement any modules with the confirmation of other participants	<ol style="list-style-type: none"> 1. Utilize automation tools to reduce unnecessary time; 2. Optimize the process;

them to some degree, the way of DevOps enhancement can provide with a more effective solution by not only automation tools, but also process optimization.

C. Long-term Sustainability

Until now, the DevOps community is constantly releasing new open-source automation tools to support DevOps. As mentioned in previous section, implementing a toolchain is the crucial point of D^2Ops functional demand. Given open-source tool's variability and diversity, how to pick tools and how to integrate tools are very difficult to settle. It's necessary to propose a stable approach for guidance.

Toward this objective, a potential solution is to encapsulate business processes that contain several tools as generic components. As a consequence, the toolchain will consist of some generic process components. Obviously, this solution can not only keep process features of toolchain, but also improve the flexibility for future changes.

IV. A REFERENCE PROCESS MODEL FOR D^2Ops

A. Participants Analysis

Recently, a controversial role called *DevOps Engineer* catches people's attention. DevOps engineers are required to possess the ability of development and operation. Sometimes they should also play a role of QA engineer. Consequently, it becomes an emerging development pattern that DevOps engineers need to do everything in development process.

But such pattern is harmful for DevOps[18][19]. For data analytical system, it requires new features about data analysis continuously. Due to the heavy business pressure, DevOps engineers can hardly make overall arrangement. On the contrary, a clear distinction in roles may improve efficiency. Thus the participants of D^2Ops process are suggested to be divided into six roles, and responsible for different business layers.

- *Data Engineer* - They are responsible for data governance and data management.
- *Data Scientist* - They should construct credible data models in different application scenarios.
- *Front-end Software Developer* - Besides providing with interactive user interface, in this case, they are more focused on the visualization of data models.
- *Back-end Software Developer* - They take charge of fulfilling the system skeleton and stable program modules, including the implementation of data model.

- *Integration Engineer* - They ought to cope with operation issues, such as installation, integration and deployment.
- *Quality Assurance Engineer* - They are responsible for discovering latent system issues before release.

B. D^2Ops Process Model

Software development is a complex process that involves with various business activities. In Fig. 1, D^2Ops process model outlines participant's activities and their collaborations. Most of participants have to cope with both of *Dev* and *Ops* works, and several activities are labeled in the figure. Here *Dev* is a generalization concept which represents the works except operations.

The root cause that leads to so intricate activity collaborations is the premised constraints (see Section III). For example, data scientists must apply data models to real data in the isolated production environment. Only this way can really verify correctness. However, the activity probably requires data scientists to do some server operations.

As we emphasized in Section III, it's more beneficial to combine process with automation tools. In order to improve the efficiency of D^2Ops process model, we should make it an integrated automation toolchain.

C. Generic Process Components

According to the sustainability demand of D^2Ops , generic process components are very important for implementing a stable DevOps process model. In the past, several researchers have done similar studies on it. Wettinger et al.[17] thought that DevOps artifacts should be divided into two major classes: Node-centric artifacts and Environment-centric artifacts. Babar et al.[8] thought that DevOps model are composed of production management, development, automated testing, ongoing deployment and operational support.

But Wettinger's solution was proposed from a perspective of deployment artifacts, and didn't consider the whole DevOps process. While although Babar took process into account, he didn't consider concrete businesses of data analytical system. Therefore, these solutions are not so suitable for D^2Ops .

Based on the D^2Ops process model, we designed a series of generic process components. Each component is composed of several automation tools that are substitutable. They are designed to solve the given process demands, and able to be the selection criteria for specific automation tools.

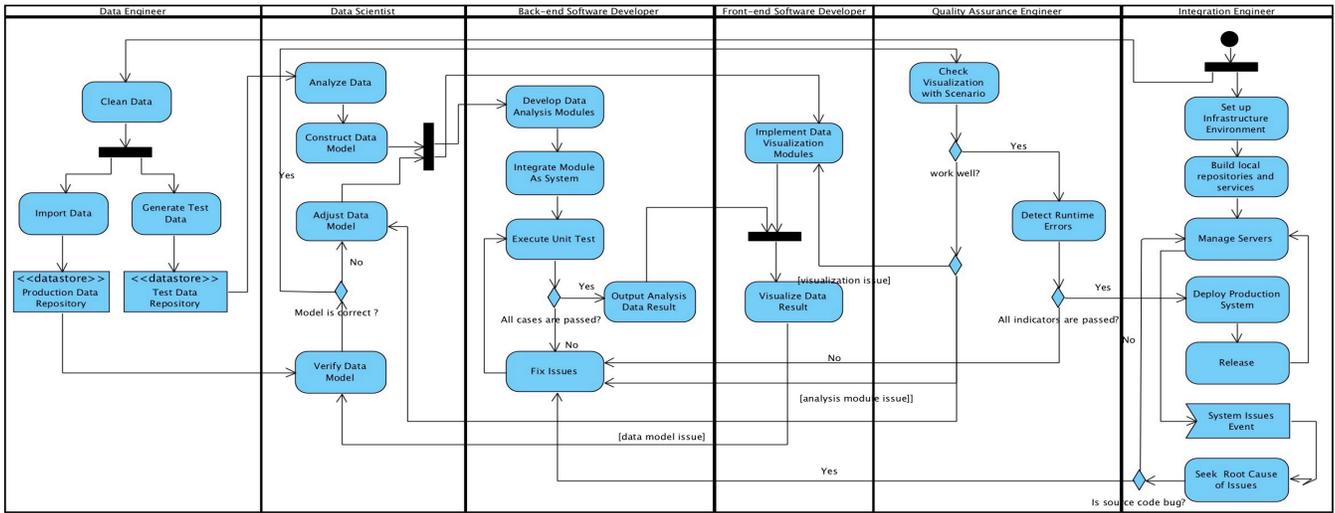


Fig. 1: D^2Ops Process Model

- *Server Management Component* - A component that solves the obstacles of server management. Such as configuration, installation and monitoring.
- *Data Processing Component* - A component that is responsible for data governance. It needs to integrate data from different sources into uniform format and make it meet business requirements. Its work also includes the cleaning of useless data and the storage of processed data.
- *Development Environment Component* - A component that encapsulates development environment. It provides both Dev team and QA team with a unified environment to avoid problems caused by environment difference.
- *Version Control Component* - A component that stores and manages version histories of source codes and artifacts that can recover data models. Its hook mechanism is an important foundation for automatic build.
- *CI Build Component* - A component that takes charge of integrating newcomer source codes. It's a process collection of compiling, building and testing source codes to produce a function module[20].
- *Portability Component* - A component that encapsulates program modules and relevant runtime environment. It's portable and concentrates on solving the troubles caused by isolation restrictions.

V. ESSENTIAL WORKFLOWS OF D^2Ops PROCESS MODEL

In Fig.2, we demonstrate a reference facility for D^2Ops process model on the basis of generic process components. It concentrates on fast feedbacks and portable deployment by automation tools. Parts of available open-source tools are presented in TABLE II. In the perspective of functionality, the facility can be divided into three workflows roughly.

A. Production Development Workflow

In recent years, Docker, an open-source implementation of operating system-level virtualization, is rapidly gaining mind-

share amongst developers[21]. The rise of Docker becomes a strong support on the assumption of *Portability Component*. As a virtual container, it endows users with the capability to package program modules and their runtime environment. Its features, encapsulation and portability, make it suitable for *Development Environment Component* as well.

Due to the isolation situation, the key goal of this workflow is to generate stable *Portability Components* for the production system and data model modules. As is shown in Fig. 2, there should be two repositories in *Version Control Component*. One is for data model artifacts, and another one is for the source codes of production system. When data scientists commit their modification for data models, a feedback from *Version Control Component*, like email, will be sent to developers quickly. Then front-end developers and back-end developers can start to implement the data models and the relevant visualization immediately. Once they also commit source codes by *Version Control Component*, it will trigger hooks set for *CI Build Component*, then run automated testing scripts. Provided that a testing case was failed, *CI Build Component* would send feedbacks to notify both developers and QA engineers. Otherwise, the newcomer source codes will be integrated into lasted source codes and built as *Portability Component*. Afterwards, it's time for QA engineers to do some testing procedures. If everything researches the standard, integration engineers can deploy it to the isolated server. By this workflow, finally, the production

TABLE II. Available tools for generic process components

Component	Open-source automation tools
Server Management Component	Puppet, Chef, Ansible, SaltStack
Data Processing Component	Pentaho Kettle, Talend, KETL
Development Environment Component	Virtual Box, Vagrant, Docker
Version Control Component	<i>Client:</i> Git, Subversion, Mercurial <i>Server:</i> GitLab, Gogs, VisualSVN, Phabricator
CI Build Component	Jenkins, BuildBot, Travis CI
Portability Component	Docker

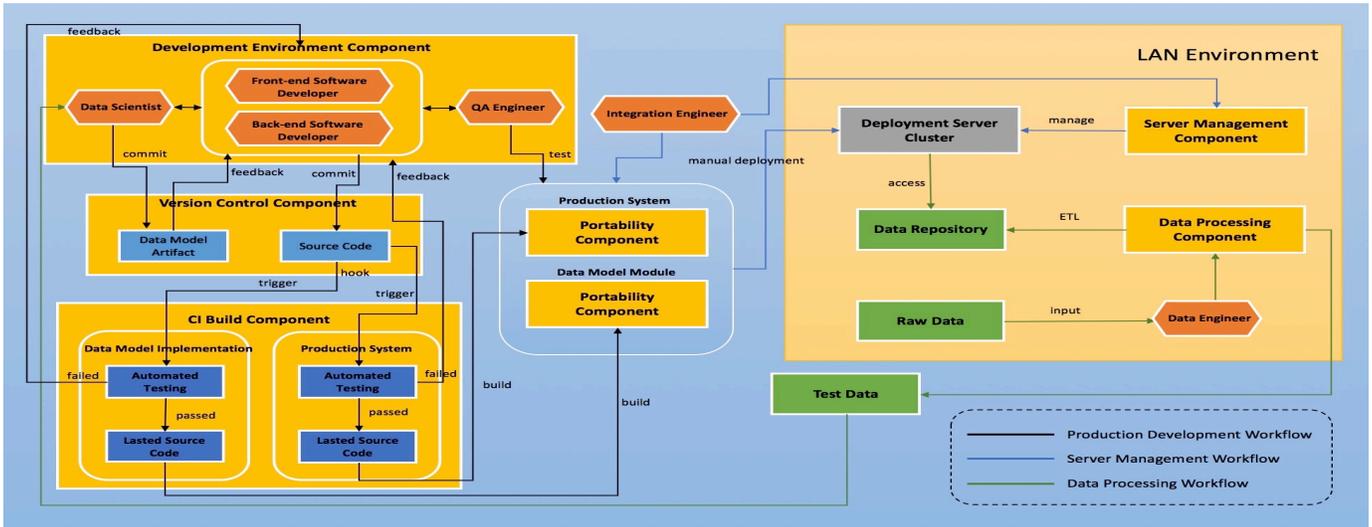


Fig. 2: Reference Facility Based on Generic Process Components

system can run on the server successfully. Meanwhile, data scientists are able to verify their data models.

This workflow is the most critical part in the D^2Ops process model. Its continuous integration way can reduce many unnecessary labor costs and communication costs.

B. Server Management Workflow

As is well known, deployment servers are the most foundational infrastructure. In order to manage server cluster intensively, several open-source automation tools were released. Their core concept is to maintain automation scripts in the master machine, and remotely control minion machines to execute corresponding commands. It's so-called "Infrastructure as Code"[2][22]. By this way, integration engineers can maintain massive servers at the same time. It reduces troubles brought by repetitive work. Moreover, it's also able to monitor server's running status and obtain relevant information constantly.

Nowadays many automated tasks for server management depend heavily on the Internet services. But as illustrated in TABLE I, the constraint, *Network Accessing Control*, disables those services. As a result, integration engineers have to handle all management details. To continue utilizing the feature of those services, a better solution is to publish local services. It requires integration engineers to confirm the needed repository sources on the Internet, and make duplications in local servers.

C. Data Processing Workflow

This workflow connects raw data with production system and data models. It makes the D^2Ops process model become a complete streamline. The workflow mainly focuses on the automation implementation of data governance. When new raw data come into *Data Processing Component*, it will activate this component to clean up dirty data and execute transformation actions automatically. Finally, the processed data will be stored in the *Data Repository*. Meanwhile, the corresponding test data will also be generated. At that time, data scientists can use them to construct data models.

VI. CASE STUDY

As criminal activities are increasingly rampant, traditional ways are unable to satisfy the requirement of solving a crime. Nowadays, utilizing a data analytical system becomes a solid choice for relevant organizations gradually.

We have been appointed to develop an intelligence reconnaissance system called *IAGraph*, which is a kind of data analytical system. Based on the existing criminal-related data, it mainly focuses on the analysis about latent criminal detection. The existing data include criminal records, trip informations, monitoring data and so on. There is no doubt that no matter which of them is leaked, the consequence will be extremely serious.

Under the instruction of D^2Ops solution proposed in this paper, we built this system successfully. Fig. 3 demonstrates our implementation of D^2Ops facility in this case. Each process component is assigned with explicit automation tools, including Vagrant, GIT, Gogs, Jenkins, Docker, Puppet and Pentaho Kettle. Given the possibility of platform migration in the future, all the selected tools are cross-platform.

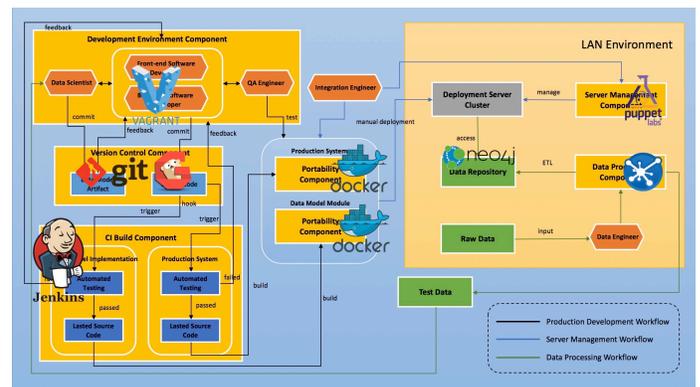


Fig. 3: Reference Facility Implementation

CODE SNIPPET 1: Jenkins Shell Script

```
set -e
CID=$(docker ps | grep "tjsse/iagraph" | awk '{print $1}')
if [ "$CID" != "" ]; then
  docker stop $CID
  docker rmi tjsse/iagraph
fi
rm Dockerfile
wget http://192.168.1.116:3000/DockerfileHub/src/master/IAGraph/Dockerfile
docker build -t="tjsse/iagraph" -f="Dockerfile" .
docker run --name iagraph_run -p 3000:9000 tjsse/iagraph activator run
CCID=$(docker ps | grep "tjsse/iagraph" | awk '{print $1}')
docker commit $CCID ieven/iagraph:deploy
docker save -o IAGraph.tar tjsse/iagraph:deploy
```

CODE SNIPPET 2: Dockerfile Script

```
FROM ingensi/play-framework:latest
MAINTAINER iceplus <zhengjiabin.tj@gmail.com>
ADD ./IAGraph /app
```

Because of the limited space, here we only describe the creation steps of Docker image in detail. The technical key point is to utilize the mechanism of *Git Hook*. In this process, a Jenkins plugin identified by *build-token-root* should be installed necessarily. It endows Jenkins with the ability to build project remotely by HTTP request. After setting an authentication token using this plugin, we configured a related Webhook for git *push* event in Gogos repository. Once developers push their codes to Gogos repository, the Webhook would trigger and send a *POST* request to the specified URL. Then Jenkins would execute the shell script shown in CODE SNIPPET 1 to generate a portable Docker image *IAGraph.tar*. In addition, CODE SNIPPET 2 illustrates the corresponding Dockerfile that defines specific instructions of image creation. Finally, integration engineers could dispatch *IAGraph.tar* to all servers by Puppet, and execute a command “*docker load < IAGraph.tar*” for loading the docker image to Docker platform. Then the deployment work came to an end.

VII. CONCLUSION AND FUTURE WORK

In this paper, we have concentrated on exploring DevOps for data analytical system. After thoroughly analyzing the *D²Ops* demands including various constraints, we proposed a process model for *D²Ops*. Furthermore, we also designed six generic process components to support this model.

Our research achievement takes the constraints of system characteristics as its premise and is able to integrate open-source automation tools as a toolchain effectively. Moreover, it can also be regarded as a reference case. Obviously, the achievement is able to cope with the mentioned challenges when to apply DevOps for data analytical system.

But there are several shortages in our research. Due to the lack of DevOps criteria, we can hardly quantify the effectiveness of *D²Ops*. Moreover, in order to clarify the relation among participant activities, many minor issues were neglected selectively.

For the purpose of improving *D²Ops*, we will be dedicated to solve these shortages in the future. The specific points are presented as follows.

- Conduct abundant relevant investigations and propose a set of evaluation criteria

- Consider more detailed activities
- Carry out *D²Ops* solution in more data analytical systems to collect feedbacks

REFERENCES

- [1] M. Stillwell and J. G. Coutinho, “A devops approach to integration of software components in an eu research project,” in *Proceedings of the 1st International Workshop on Quality-Aware DevOps*. ACM, 2015, pp. 1–6.
- [2] M. Huttermann, *DevOps for developers*. Apress, 2012.
- [3] M. Walls, *Building a DevOps culture*. O’Reilly Media, Inc., 2013.
- [4] S. K. Bang, S. Chung, Y. Choh, and M. Dupuis, “A grounded theory analysis of modern web applications: knowledge, skills, and abilities for devops,” in *Proceedings of the 2nd annual conference on Research in information technology*. ACM, 2013, pp. 61–62.
- [5] L. E. Lwakatare, P. Kuvaja, and M. Oivo, “Dimensions of devops,” in *Agile Processes, in Software Engineering, and Extreme Programming*. Springer, 2015, pp. 212–217.
- [6] N. Ferry, F. Chauvel, H. Song, and A. Solberg, “Continuous deployment of multi-cloud systems,” in *Proceedings of the 1st International Workshop on Quality-Aware DevOps*. ACM, 2015, pp. 27–28.
- [7] D. E. Weeks, “Devops and continuous delivery reference architectures,” <http://www.slideshare.net/SonatypeCorp/nexus-and-continuous-delivery>, <http://www.slideshare.net/SonatypeCorp/devops-and-continuous-delivery-reference-architectures>, accessed March 16th, 2016.
- [8] Z. Babar, A. Lapouchnian, and E. Yu, “Modeling devops deployment choices using process architecture design dimensions,” in *The Practice of Enterprise Modeling*. Springer, 2015, pp. 322–337.
- [9] M. Corp, “Devops and application lifecycle management,” <https://www.visualstudio.com/en-us/features/alm-devops-vs>, accessed March 16th, 2016.
- [10] I. Corp, “Enabling business agility: Devops for the enterprise,” <http://www.ibm.com/ibm/devops/us/en/>, accessed March 16th, 2016.
- [11] A. Corp, “Enabling business agility: Devops for the enterprise,” <https://aws.amazon.com/cn/campaigns/emea-devops/>, accessed March 16th, 2016.
- [12] D. Bruneo, T. Fritz, S. Keidar-Barner, P. Leitner, F. Longo, C. Marquezan, A. Metzger, K. Pohl, A. Puliafito, D. Raz *et al.*, “Cloudwave: Where adaptive cloud management meets devops,” in *Computers and Communication (ISCC), 2014 IEEE Symposium on*. IEEE, 2014, pp. 1–6.
- [13] J. Kim, C. Meirosu, I. Papafili, R. Steinert, S. Sharma, F.-J. Westphal, M. Kind, A. Shukla, F. Németh, and A. Manzalini, “Service provider devops for large scale modern network services,” in *Integrated Network Management (IM), 2015 IFIP/IEEE International Symposium on*. IEEE, 2015, pp. 1391–1397.
- [14] S. Elliot, “Devops and the cost of downtime: Fortune 1000 best practice metrics quantified,” *International Data Corporation (IDC)*, 2014.
- [15] V. P. R. Kumar and V. Babu, “Devops-a review,” *IMAGE AND VIDEO PROCESSING*, p. 32, 2012.
- [16] A. M. Christie, *Software process automation: the technology and its adoption*. Springer Science & Business Media, 2012.
- [17] J. Wettinger, U. Breitenbücher, and F. Leymann, “Standards-based devops automation and integration using toasca,” in *Proceedings of the 2014 IEEE/ACM 7th International Conference on Utility and Cloud Computing*. IEEE Computer Society, 2014, pp. 59–68.
- [18] I. BJELAJAC, “Devops is a culture, not a job description,” <http://devana.rs/blog/devops-is-a-culture-not-a-job-description/>, accessed March 16th, 2016.
- [19] J. Knupp, “How ‘devops’ is killing the developer,” <https://jeffknupp.com/blog/2014/04/15/how-devops-is-killing-the-developer/>, accessed March 16th, 2016.
- [20] C. A. Cois, J. Yankel, and A. Connell, “Modern devops: Optimizing software development through effective system interactions,” in *Professional Communication Conference (IPCC), 2014 IEEE International*. IEEE, 2014, pp. 1–7.
- [21] J. Fink, “Docker: A software as a service, operating system-level virtualization framework,” *Code4Lib Journal*, vol. 25, 2014.
- [22] D. Spinellis, “Don’t install software by hand,” *Software, IEEE*, vol. 29, no. 4, pp. 86–87, 2012.

Criminal Network Analysis with Interactive Strategies: A Proof of Concept Study using Mobile Call Logs

Peng ZHOU*, Yan LIU†, Mengjia ZHAO‡
School of Software Engineering, Tongji University
Shanghai, China

Email: *1435855@tongji.edu.cn †yanliu.sse@tongji.edu.cn
‡1434319@tongji.edu.cn

Xin Lou§

Shanghai iEven Information Technology Co.,Ltd.
Shanghai, China

Email: §steven.lou@ieven.com.cn

Abstract—The communication data are becoming increasingly important for criminal network analysis nowadays, this data provides a digital trace which can be regarded as a hidden clue to support the crack of criminal cases. Additionally, performing a timely and effective analysis on it can predict criminal intents and take efficient actions to restrain and prevent crimes. The primary work of our research is to suggest an analytical process with interactive strategies as a solution to the problem of characterizing criminal groups constructed from the communication data. It is expected to assist law enforcement agencies in the task of discovering the potential suspects and exploring the underlying structures of criminal network hidden behind the communication data. This process allows for network analysis with commonly used metrics to identify the core members. It permits exploration and visualization of the network in the goal of improving the comprehension of interesting microstructures. Most importantly, it also allows to extract community structures in an appropriate level with the label supervision strategy. Our work concludes illustrating the application of our interactive strategies to a real world criminal investigation with mobile call logs.

Keywords—criminal network analysis; interactive strategy; network measure; community detection;

I. INTRODUCTION

Communication data are widely used in criminal network analysis to understand direct relationships and identify implicit connections. Many efforts have been devoted to leverage those data in criminal community detection, connection strength evaluation, microstructure discovery, and suspect identification. However, the skill set required for criminal network analysis (CNA) is complex and diverse, such as the application of empirical study and domain knowledge into the data preprocessing, criminal investigation knowledge, intelligence analysis experience, network visualization layout technique in different network scale, and social network analysis knowledge, which leads mismatching of expectation and reality on the power of data analytics.

We proposed an interactive analysis process by observing practical workflows which involving detectives, intelligence officers, data engineers, data scientist, and domain experts, combined with the technique of social network analysis and

machine learning. This process is divided into three phases according to important results in each phase, namely, *i) network construction*: the generation of network structure, *ii) metric design*: the core nodes and relations, *iii) structure observation*: structures extraction in different levels. In each phase, the process adopted interactive strategies in order to formulate and assess hypotheses in a rapid, iterative manner—thus supporting exploration Criminal Networks (CN) with the pace of human thought.

The interactive analysis process is our chosen way to explore the CN with a case study using the mobile call log. In the construction phase, we build the CN by data preprocessing, data cleaning and data extraction, which contains 7840 nodes and 103043 links, and then three layout are introduced to configure and depict the CN. Metric analysis phase illustrated the adoption of some classic metrics in social network analysis seeking to identify the key group members, and the evaluation of these metrics with corresponding result. Finally, in the structure analysis phase, Fast Unfolding is used as our detection algorithm to analyse CN interactively and rapidly. Additionally, we proposed a strategy — label supervision to control the level of the structure extracted.

Our contributions contain: 1) suggesting a generical analytical process for Criminal Network Analysis. It can be divided into three phases. 2) proposing interactive strategies to the analytical process, such as using various visualization layouts to configure the network, reinterpreting different measures from the social network domain to the criminal network domain, and controlling the community structure level with label supervision strategy. 3) conducting a proof of concept study using mobile call logs.

The paper is organized as follows. Section II describes related works about CNA. Section III illustrates how we proposed the interactive analytical process and introduced three visualization layouts for CN. In the following, Section IV, section V and section VI demonstrates the three phases separately with an example of Criminal Network being constructed, analyzed with metrics and explored with detection algorithm. The final section, section IIV, concludes the future

work and a conclusion according to our work.

II. RELATED WORK

In this section, an overview of previous works in the domain of criminal network analysis has been firstly provided, and then comes to the concept of community structure and structure detection method. Finally, we present a comprehension on the two relevant works about its advantages and limits.

A. Criminal Networks Analysis

In the last thirty years, many efforts have been used in order to analysis the Criminal Networks in a more intelligent way. One of the most important research in the CNA domain is due to Malcolm Sparrow [1], who summarized four features of the Criminal Network, namely, i) limited dimension — CNs are often composed of few thousands entities; ii) incomplete information — criminal networks are unavoidably incomplete and erroneous; iii) undefined border — it is difficult to pick out all the relations of each entity; and, iv) dynamism — new relations always indicate constant evolution of network structure.

Fortunately, this contribution lead to a trend that researchers tried to analyze Criminal Networks with the techniques in the domain of the social network analysis. For instance, Baker and Faulkner [2] studied illegal networks in the field of electric plants and Klerks [3] concentrated on criminal organizations in the Netherlands. Silke [4] and Brannan et al. [5] acknowledged a slow growth in the terrorism network and examined state of the art in criminal network analysis. Arquilla and Ronfeldt [6] summarized previous researches and proposed the concept of Netwar with its application to terrorism.

However, in 2006, a popular work by Valdis Krebs [7] applied network analysis in conjunction with network visualization theory to analyze the 2001-09-11 terrorist attacks. This work represents a starting point of a series of academic papers in which social network analysis methods become applied to a real-world case, differently from previous work where mostly toy models and fictitious networks were used. Krebs paper inspired further research in network analysis for the design of better SNA applications to support intelligence agencies in the fight against terror, and law enforcement agencies in their quest fighting crime.

B. Community Structure and Community Detection

Community structure is one of the most common characteristics in the study of networks [8], which refers to the occurrence of groups of nodes in a network that are more densely connected internally than with the rest of the network [9]. There is one widely adopted concept to investigate the quality of this structures in the network called network modularity, which can be expressed as follows: let consider a network, represented by $G = (V, E)$, which has been partitioned into m communities; its corresponding value Q of network modularity is defined as:

$$Q = \sum_{s=1}^m \left[\frac{l_s}{|E|} - \left(\frac{d_s}{2|E|} \right)^2 \right] \quad (1)$$

assuming l_s the number of links between nodes belonging to the s_{th} community and d_s is the sum of the degrees of the nodes in the s_{th} community. High values of Q indicate high values of l_s for each discovered community, yielding to communities internally densely connected and weakly coupled among each other. The process of detecting community structures in a network called community detection, and it still is regarded as a computationally difficult task. However, several methods for community detection have been proposed and applied with varying extents of success. Such as minimum-cut method [10], hierarchical clustering [11], Girvan-Newman algorithm [12], modularity maximization and clique based methods [13].

Lots of works shown that community detection is proved as a powerful tool to analyze the structure in the criminal networks. Emilio et al. [14] employed Girvan-Newman algorithm and a variant based on modularity optimization called Newmans algorithm to detect and explore the community structures in the CNs reconstructed from phone call logs. Hamed Sarvari et al. [15] performed a large scale analysis with clique based methods to find patterns and substructures of that network based on a publicly leaked set of customer email addresses. However, these studies and early researches somehow neglected the importance of network visualization and the interactions during the analysis process, laying emphasis on aspects related more to statical network characterization. In our research, we stress these twofold by introducing various visualization layouts and adopting crucial interactive points.

III. INTERACTIVE STRATEGIES

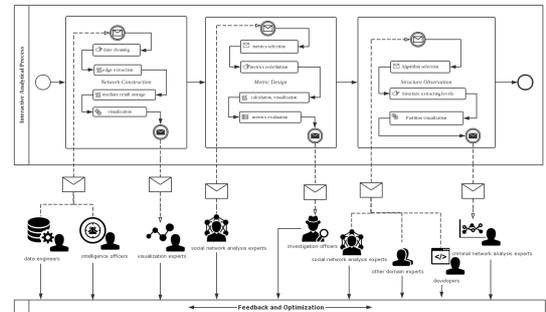


Figure 1. The interactive analytical process

At present, the task of analysing Criminal Networks can not be finished purely relying on the computational analysis or manual analysis. We examined all the analytical process in these works and we found that almost all the processes about CNA are interactive process between humans and computers, but the interactive strategies are weak, limited and even not involved obviously and deeply. In order to gain better understanding of the process with some strong interactive strategies during the CNA, we followed 5 investigations based on a real world criminal cases and also observed a general pattern of work shared by most police officers and intelligence analysts.

A. Interactive Analysis Process

After doing induction and summary on the workflow, we concluded that the process of CNA is a human-interaction process, and proposed an interactive analytical process for CNA. As is shown on the Figure 1. the process can be mainly divided in to three phases according to each phase's promising results, namely, *i) network construction*: the generation of network structure, *ii) metric design*: the core nodes and relations, *iii) structure observation*: the extraction of organization structure. And this process is supposed to assist investigator to analysis Criminal Network in an interactive way.

In the first phase, we clean data interactively depending on the empirical rules suggested by intelligence officers and data engineers, extract the edge table from those data, provide a format of intermediate result storage and finally perform network visualization based on the layouts selected by visualization experts. In the second phase, some metrics are introduced by SNA experts from the domain of SNA into the domain of CNA. And there is also need to reinterpret them before employing in the context of CN, after which we might be able to obtain the ranking results. Additionally, we perform various configurations on network with each entities corresponding metric value. At the end of this phase, a certain evaluation is defined to suit for our CN to examine the quality of different metrics in consultation with investigation officers. The crucial point related to interaction in this phase is twofold, one is metrics selection which can measure different centrality and influence of each node, another is that we need to configure many kinds of layouts in consideration of the scale of the CN. The following phase is structure analysis of the CN. First, we need to choose a detection algorithm, taking the features of the CNs constructed during the phase one into consideration. For example, different scale might need different methods, different complexity likewise need different method. So we might need the support of SNA experts, other domain experts and programming developers. Then ascertain the extracting structure level to amplify or narrow down graph interests. Finally, a partition visualization results is produced enabling to examine the quality of the structure and even find out the hidden knowledge. Note that the three phase is an integrated whole. The previous stage outputs is regarded as the next phase of the input, so each phase results would interfere with the next phase results. Consequently, we suggest another stage — feedback and optimization to ensure the effective communication between various domain experts and optimization of the whole interactive process.

B. Data Set

For better illustrating the following works, here we present a brief introduction of the dataset. A brief description of the data set is shown in Table I. With a preliminarily discovery on the dataset, we find that the dataset contains 73 detailed phone call tickets, among them containing 25 known suspect and 48 persons involved with this investigation case. All phone call tickets are all in the period from 2014.6 to 2015.6 summing

TABLE I. Date set summary

Detailed ticket	Known suspect profile	Time period	Total call logs
73	25	2014.1 2015.6	1016833

up all the phone call logs from all the call tickets, we got totally 1016833 logs.

IV. NETWORK CONSTRUCTION

In order to assure the quality of the network construction, we proposed a generical interactive workflow to support this phase task. This workflow shows how the phone call logs are preprocessed, transformed, extracted and constructed into a network.

A. Data Preprocessing and Network Construction

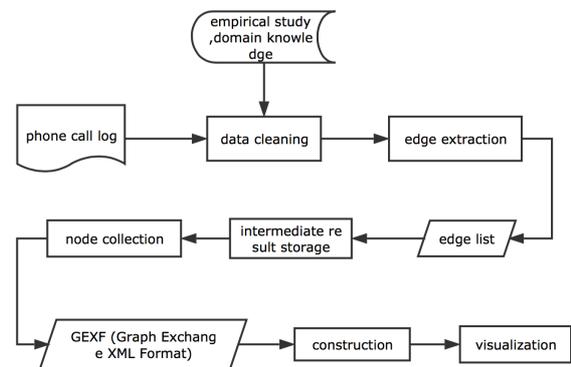


Figure 2. The data preprocessing workflow

Performing a good preprocessing on the dataset can get rid of the noise as much as possible, and it permits the network structure to be constructed easily without cost of quality. It allows the network metrics to depict a member more exactly and make visualization closer to the nature of the criminal communication network. Additionally, it likewise can reduce the following analytical errors and the workload of the structure analysis phase. Owing to the great importance of data preprocessing, we think highly of this stage.

The data preprocessing stage was shown on the Figure 2. The data cleaning comes firstly to our work, the empirical study and domain knowledge that we used as follows:

a) eliminate the call logs containing bank notifications, communication providers and other public service providers. For instance, the number 10086 is a communications service providers in China, and 95595 is the call number of Agricultural Bank of China. *b)* remove redundant logs produced by peer to peer calling. When one entity calls another entity, both of their phone detailed tickets will generate almost the same log with a certain time difference and different call types that one is calling and another is called. *c)* wipe off the call logs whose call duration is zero. This type of logs indicate that is not a successful connection and should not be considered in the following works.

After data cleaning, the workflow extracts the data with few noise to build the edge list of the network. In this step we need

to change call type according to the corresponding call type in order to identify the source nodes and the target nodes. When finishing edge extracting, our work chooses a intermediate result storage. Then we write a python script to collect the nodes to build the node set and importing the nodes set and edge list by the python-igraph package and constructing our CNs, finally we export GEXF file in order to visualize by a software called Gephi. At length we construct a burglary criminal network shown on the figure 3 (a), which has totally 7840 nodes and 1016833 links.

V. METRIC DESIGN

In order to exactly capture the influence of core members and its roles playing in the criminal organizations, we introduce a series of measures from the domain of Social Network Analysis and interpret them in the context of our certain CN. After calculating these metrics, we render the network view with different color intensity and size based upon the metric value of each entity and spotlight certain nodes.

A. Network Measures Reinterpretation in CNs

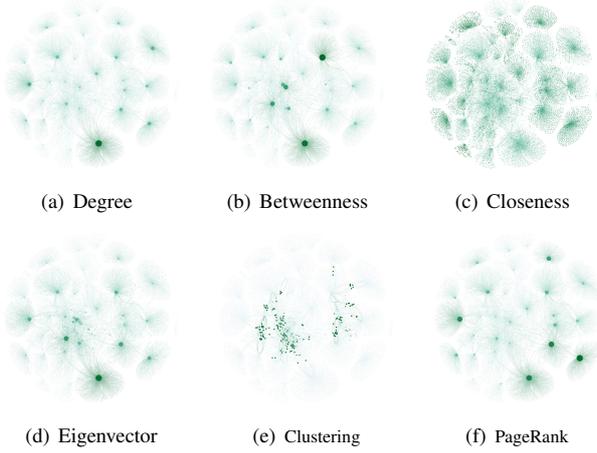


Figure 4. Network metrics

Degree centrality measures the activity and influence by calculating the number of direct relations a node has. It is convinced that a call number with a high degree could be regarded as a hub which act as an import information channel in the communication network and a node with high activity. It is defined as:

$$Dgr_i = \frac{d_i}{S-1} = \frac{\sum_{i \in M} m_{ij}}{S-1} \quad (2)$$

where d_i is the directly links to other call numbers of a call number i , and m_{ij} is the ij_{th} element of the adjacency matrix M and S is the sum of the whole call numbers. $S-1$ is the normalization factor.

Betweenness centrality is another indicator to measure nodes centrality which equals the number of the shortest paths from all nodes to all other nodes which pass through that node.

In the context of CN, a call number with high betweenness can be regarded as broker of messages, indicating the great importance of this call number in the information transfer. Most importantly, this kind of call numbers usually connects two or more densely call number clusters, removal of them might lead to the destabilization, abrupton and even destroy of the CNs. The betweenness centrality of a call number (i) is redefined as:

$$Btw_i = \frac{\sum_{j < k \in N} \frac{p_{jk}(i)}{p_{jk}}}{(S-1)(S-2)} \quad (3)$$

where p_{jk} is the total number of the shortest paths from call number j to call number k and the $p_{jk}(i)$ is the number of those paths that pass through call number i , the $(S-1)(S-2)$ is the normalization factor.

Closeness centrality is the reciprocal of the sum of the shortest paths between that node and all other nodes in the network. In the analysis of our CN, a high closeness indicates this call number can reach majority of the other nodes easier and faster. The closeness centrality of a call number (i) is redefined by the expression:

$$Clsn_i = (H_i)^{-1} = \frac{S-1}{\sum_{j \in N} d(i, j)} \quad (4)$$

where $d(i, j)$ is the distance between node i with node j , H_i is the normalized distance.

Eigenvector centrality defines another centrality with the consideration of the importance of their neighbours. In the framework of our CN, a call number with high eigenvector centrality means that this call number can reach a group of other call numbers easily and quickly. The eigenvector centrality of a node (i) is redefined as:

$$Eign_i = \frac{1}{\alpha} \sum_{j \in L(i)} m_j = \frac{1}{\alpha} \sum_{j \in N} a_{ij} x_j \quad (5)$$

where $L(i)$ is the direct link set of call number i , α is a constant, and a_{ij} is the ij_{th} element of the adjacency matrix M .

Clustering coefficient is a measure of the degree to the aggregation of nodes in a graph. In the context of our CN, a call number with high clustering coefficient means a high likelihood of that the direct links of the given call number can reach each other. It is redefined as:

$$Clst_i = \frac{|e_{jk}|}{l_i(l_i-1)} \quad (6)$$

where e_{jk} is the link existing in the neighbours of call number i and l_i is the number of direct links of the call number i .

PageRank is a variant of the eigenvector centrality measure in some degree. Our work employ this measure to the importance of a certain call number globally. It is redefined as:

$$Page(i) = (1-f) + f * \sum_{j \in H(i)} \frac{P_i(j)}{L_j} \quad (7)$$

TABLE II. Different network metrics of the known suspect

Dgr	Btw	Clsn	Eigvt	Clst	PgRk
9699*	1002*	7071	5636*	5117	7777*
0691*	9699*	1083	8662*	3229	9699*
1509*	5636*	8890	0691*	4903	8662*
7777*	2406*	5703	1002*	5064	0691*
8662*	8662*	4848	7777*	7037	7789*
8199*	4751*	7515	5676*	9868	1002*
1002*	8199*	8986	8199*	1115	8199*
2223*	6526*	9549	9699*	7777*	9250*
1359*	7777*	5362	2406*	8318	3605*
5636*	0691*	4097	2223*	5782	5636*
9250*	1509*	9891	0399*	9699*	2223*
7789*	2223*	8582	4076*	6409	1509*
3605*	7970*	3401	9250*	0691*	8391*
8391*	8391*	0206	7789*	5435	1359*
2406*	8318	1223	4494	8662*	2406*
5676*	1359*	2601	8391*	5347	6526*
7970*	5676*	6179	1509*	7071	7970*
7574*	4494	0393	1359*	8199*	5676*
6526*	7789*	3012	3605*	1083	7574*
3157*	3605*	0722	6859	8890	7603*
4076*	7588	5894	6526*	7789*	4076*
7692	9250*	3040	8033	1002*	0399*
2647*	3157*	5049	4751*	9250*	3157*
0399*	4076*	2602	2397	3605*	2647*
4751*	7574*	4892	1050	5636*	7692

where $H(i)$ are the set of call numbers directly linking to the call number i , L_j is the number of outgoing links in j and f is the damping factor.

B. Metrics Visualization and Ranking Interpretation

Our visualization results with the combination view of Fruchterman Reingold layout and Fisheye layout was shown on the Figure 4, and the color intensity and the size of the nodes both are rendered by their corresponding metric value. To check the usability of different metrics, we defined the hit rate of the known burglary suspect in the top 25 and top 100 rank as:

$$h = \frac{n}{K} * 100\% \quad (8)$$

where n is the number of known suspect in the top K ranking of the corresponding metric. The hit rate results were shown on the Table III.

To begin with, we can found that the degree, betweenness, eigenvector and PageRank have a similar and excellent layout view. After checking the hit rate of these metrics, it indicated that 96% of suspect belongs to the top 25 of the degree, 88% to betweenness, 84% to eigenvector, 96% to PageRank. Therefore in the top 100 ranking, they consequently got 100%. These four metrics were proved a valid measure for burglary group CN. But the remaining two metrics were not good enough and even no one suspect was ranked in the top 25 of closeness metric. Although the hit rate of clustering coefficient only got 60% in the top 25. After we performed a manual check on the top 100, it works well. But for closeness centrality, it still got 0% in the top 100. The most likely factor resulting in this phenomenon is the way of data collection.

In summary, the metrics selection should take the way of the data collecting into consideration ensuring that the effective

TABLE III. The hit rate of top-K ranking

Rank	Dgr	Btw	Clsn	Eigvt	Clst	PgRk
Top 25	96%	88%	0%	84%	60%	96%
Top 100	100%	100%	0%	100%	100%	100%

metrics for measuring the CNs would be adopted. In the context of burglary group phone CN, these four indicators which are degree, betweenness, eigenvector, and PageRank, are good enough to analysis the network in an effective way. The clustering coefficient might work well if we broaden the ranking scope.

VI. STRUCTURE OBSERVATION

After the metric design phase, this section is focusing on the structure analysis of the CN and is expected to select a suitable algorithm to finish the task of community extraction rapidly and interactively.

A. Fast Unfolding with Label Supervision Strategy

In the consideration of the scale of our burglary CN, which contains 7840 nodes and 103043 links, and the demands for interactive point mentioned in the section III, Fast Unfolding [16] has been adopted to detect the communities structure in burglary CN. Besides, we provide a label supervision strategy to put priori knowledge and evidence into the structure extraction. Thus, the level of the community structure can be well controlled. Finally, we visualize our community results with combination of FR layout and fisheye layout. Besides the extremely high speed, another reason why we choose Fast Unfolding algorithm is that this heuristic method provides a parameter of resolution to control the scale of the communities detected. In most situations the default value of the resolution is 1, if the value is lower, then the size of the communities detected would become smaller, which also means that it will produce more communities. If we set a higher value than the default, it means bigger communities structure and less communities. Therefore, in this way, our process can gradually increase or decrease the value of resolution in order to extract subgroups with appropriate scales in the burglary CN.

However, it is unknown when we should stop reduction or increasements of the resolution to gain the appropriate structure level. we proposed a general method based on label supervision to solve this question which can be suitable for all the detection algorithm. According to detection algorithm initial stage, it can be divided into forward and reverse strategy. If the detection is start from one community to appropriate mounts of community, it means that the process of the detection is one to more, and the label supervision should inspect the partition of the nodes labeled in the same level, we called this type forward strategy; but if the algorithm is from more to less, the label supervision should inspect the merge of the nodes in the same level. this type called reverse strategy. The following steps describe the process of the FU with the reverse label supervision strategies: 1) assign each node to a different community, for each node i , consider the neighbours j of i , put node i to its neighbour j when reaching the maximum

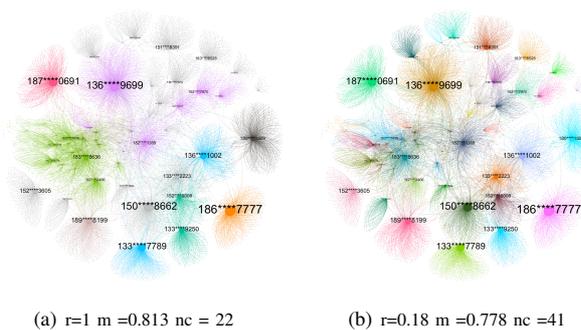


Figure 5. The left figure shows a community detection with resolution = 1 and the number of communities is 22. The right part of the figure is the situation with resolution of 0.18

positive gain of modularity 2) check the nodes labeled in the same level whether be assign into the same structure. 3) take the communities found during the step 1 as a nodes and build a new network, then back to step 1.

In the context of the CNA, when we know several entities in a same structure level, then we label this node and examine them after each phase of the Fast Unfolding working on the network till that they are arranged into different communities, and this detection can be seen as an effective analysis.

B. Structure Partition Results

The Figure 5 (a) has shown the result after performing FU on the burglary criminal network with the default resolution value, where the modularity value Q is 0.813 and the number of the communities is 22. The network was partitioned into different substructures and each communities, including nodes and edges, were rendered by different colors and the size of the node varies with the community scale. In order to gain a appropriately structural levels, then we marked the known suspects call number 136****9699 and 182*****1359 shown on the Figure 6 (a) and put this knowledge into each iteration of the Fast Unfolding algorithm with a step of 0.1 reduction of the resolution. After 83 iterations, we examined these call numbers were assigned into different communities shown on the Figure 6 (b), where the resolution is 0.18, modularity is 0.778 also a high value and the number of communities is 41. This partition result is shown on the Figure 5 (b).

An another goal of this phase is to find out potential suspects in burglary gang. In the Figure 6 (c), we can find the entity with phone number of 152****0099 which was rendered by brown kept large amounts of relation with top ranking criminal entities, which are 150****8662, 186****7777, 133****2223, 133****9250. Consequently this entity maybe a another member of the burglary group, we recommend such entities to officer for support the following investigation.

VII. CONCLUSION

In this paper, we introduce an interactive analytical process to explore the CNA with a case study using mobile phone logs. This process generally works well with the CN in our case study. In detail, the core members of the burglary group usually

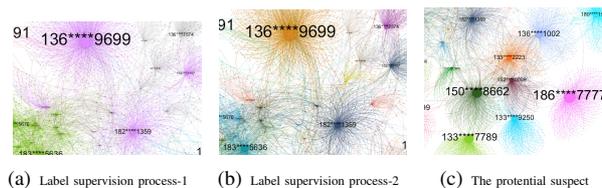


Figure 6. The sample of label supervision method

get a high ranking in network metrics, but not all measures are effective for our analysis, such as closeness centrality and clustering coefficient, and three visualization layouts helps a lot during the whole process. Most importantly, our framework can extract community structures in a appropriate level with the application of label supervision into the Fast Unfolding algorithm.

REFERENCES

- [1] M. K. Sparrow, "The application of network analysis to criminal intelligence: An assessment of the prospects," *Social networks*, vol. 13, no. 3, pp. 251–274, 1991.
- [2] W. E. Baker and R. R. Faulkner, "The social organization of conspiracy: Illegal networks in the heavy electrical equipment industry," *American sociological review*, pp. 837–860, 1993.
- [3] P. Klerks, "The network paradigm applied to criminal organizations: Theoretical nitpicking or a relevant doctrine for investigators? recent developments in the netherlands," *Connections*, vol. 24, no. 3, pp. 53–65, 2001.
- [4] A. Silke, "The devil you know: Continuing problems with research on terrorism," *Terrorism and Political Violence*, vol. 13, no. 4, pp. 1–14, 2001.
- [5] D. W. Brannan, P. F. Esler, and N. Anders Strindberg, "Talking to" terrorists": Towards an independent analytical framework for the study of violent substate activism," *Studies in Conflict and Terrorism*, vol. 24, no. 1, pp. 3–24, 2001.
- [6] J. Arquilla and D. Ronfeldt, *Networks and netwars: The future of terror, crime, and militancy*. Rand Corporation, 2001.
- [7] V. E. Krebs, "Mapping networks of terrorist cells," *Connections*, vol. 24, no. 3, pp. 43–52, 2002.
- [8] M. A. Porter, J.-P. Onnela, and P. J. Mucha, "Communities in networks," *Notices of the AMS*, vol. 56, no. 9, pp. 1082–1097, 2009.
- [9] M. Hamdaqa, L. Tahvildari, N. LaChapelle, and B. Campbell, "Cultural scene detection using reverse louvain optimization," *Science of Computer Programming*, vol. 95, pp. 44–72, 2014.
- [10] M. E. Newman, "Detecting community structure in networks," *The European Physical Journal B-Condensed Matter and Complex Systems*, vol. 38, no. 2, pp. 321–330, 2004.
- [11] A. J. Alvarez, C. E. Sanz-Rodríguez, and J. L. Cabrera, "Weighting dissimilarities to detect communities in networks," *Phil. Trans. R. Soc. A*, vol. 373, no. 2056, p. 20150108, 2015.
- [12] M. E. Newman, "Fast algorithm for detecting community structure in networks," *Physical review E*, vol. 69, no. 6, p. 066133, 2004.
- [13] T. S. Evans, "Cliques graphs and overlapping communities," *Journal of Statistical Mechanics: Theory and Experiment*, vol. 2010, no. 12, p. P12037, 2010.
- [14] E. Ferrara, P. De Meo, S. Catanese, and G. Fiumara, "Detecting criminal organizations in mobile phone networks," *Expert Systems with Applications*, vol. 41, no. 13, pp. 5733–5750, 2014.
- [15] H. Sarvari, E. Abozinadah, A. Mbaziira, and D. McCoy, "Constructing and analyzing criminal networks," in *Security and Privacy Workshops (SPW), 2014 IEEE*. IEEE, 2014, pp. 84–91.
- [16] V. D. Blondel, J.-L. Guillaume, R. Lambiotte, and E. Lefebvre, "Fast unfolding of communities in large networks," *Journal of statistical mechanics: theory and experiment*, vol. 2008, no. 10, p. P10008, 2008.

Combining Smartphone and Smartwatch Sensor Data in Activity Recognition Approaches: an Experimental Evaluation

Felipe Ramos

Federal University of Campina Grande
Campina Grande, Brazil
feliperamos@copin.ufcg.edu.br

Anne Moreira

Federal University of Campina Grande
Campina Grande, Brazil
annelorayne@copin.ufcg.edu.br

Alexandre Costa

Federal University of Campina Grande
Campina Grande, Brazil
antonioalexandre@copin.ufcg.edu.br

Reudismam Rolim

Federal University of Campina Grande
Campina Grande, Brazil
reudismam@copin.ufcg.edu.br

Hyggo Almeida

Federal University of Campina Grande
Campina Grande, Brazil
hyggo@dsc.ufcg.edu.br

Angelo Perkusich

Federal University of Campina Grande
Campina Grande, Brazil
perkusic@dee.ufcg.edu.br

Abstract—Activity recognition has been widely studied in ubiquitous computing since it can be used in several application domains, such as fall detection and gesture recognition. Initially, works in this area were based on research-only devices (body-worn sensors). However, with advances in mobile computing, current research focuses on mobile devices, mainly, smartphones. These devices provide Internet access, processing, and various sensors, such as accelerometer and gyroscope, which are useful resources for activity recognition. Therefore, many studies use smartphones as data source. Additionally, some works have already considered the use of wristbands and specially-designed watches, but fewer investigate the latest marketable wearable devices, such as smartwatches, which are less intrusive and can provide new opportunities to complement smartphone data. Moreover, for the best of our knowledge, no previous work experimentally evaluates the impact caused by the combination of sensor data from smartwatches and smartphones on the accuracy of activity recognition approaches. Therefore, the main goal of this experimental evaluation is to compare the use of data from smartphones as well as the combination of data from smartphones and smartwatches for activity recognition. We evidenced that the use of smartphone and smartwatch data combined can increase the accuracy of activity recognition.

Keywords—Activity Recognition; Smartwatch; Smartphone; Wearable Devices; Experimental Evaluation.

I. INTRODUCTION

Activity Recognition (AR) consists in the identification of human physical activities (e.g., walking, running, etc.) [10]. The study of this issue is quite relevant, since AR approaches can be applied to several application domains, for example for physical activities and health monitoring, detection of falls, home automation, advertisement delivery, and social networks based on daily activities [10]. Several types of sensors can be used in the recognition process, such as GPS, gyroscope, and mainly, accelerometer [8].

In the early stage, works in this area were based on devices specifically designed for this task [19], which are usually very

intrusive, or even huge and heavy to be carried by users. Thus, despite their positive results in terms of accuracy, these studies present limitations to be applied in realist environments. In the last years, AR has become a topic of interest for mobile and ubiquitous computing researchers [10], [5], especially due to the significant growth in the number of mobile devices available to users, which incorporate several types of sensors, such as gyroscope and accelerometer [8]. Thus, the latest works in the AR field have been carried out based on these less intrusive devices (e.g., smartphones [8], [10], [17], [19]), which achieve good results as data collection tools for activity recognition.

Recently, other mobile devices - wearable devices - are attracting the attention of the area, since they have reduced size and reasonable computational power [16]. Among the marketable wearable devices that can be used as data collection tool, stand out the smartwatches, because they are cheap and nonintrusive devices [13], can be worn 24 hours a day and be water resistant [2], and generally their battery life is more durable than smartphone ones [3]. Thus, they allow a less intrusive way to monitor physical activities and, consequently, the development of innovative applications [11], such as unsafe driving detection [9], the monitoring of user daily activities [16], and the assistance to people with visual impairments [13].

As can be seen, many works testify the applicability of the latest mobile devices (mostly, smartphones) in activity recognition, for example in [8], [19], [17], [7], [15]. However, other studies have already considered the use of body-worn sensors (i.e., sensors attached to user bodies) to complement smartphone data [7], aiming to obtain better results. But these body-worn sensors generally reduce the applicability of studies, since they are intrusive to be used in standard environments.

But, despite their applicability, to the best of our knowledge, there is no previous work that experimentally evaluated the impact on the accuracy of activity recognition with data from smartwatches combined with smartphone data. Because each device collects information from a different perspective,

the combination of their data can improve the accuracy on activity recognition. For instance, smartwatches are usually located at the wrist of users, and smartphones are typically located at their front pocket. For an activity recognition scenario such as driving, data from smartwatches seem to be more useful to recognize this activity. For example, by using only smartphone sensor data as input, a machine learning classifier could not distinguish if a user is the driver or the passenger.

Therefore, the main goal of our work is to perform an experimental evaluation to compare the use of data from smartphones as well as the combination of data from smartphones and smartwatches for activity recognition. We perform this experiment with thirteen participants, mainly undergraduate students, aging between 20 to 35 years old. To accomplish that, we simultaneously collect accelerometer data from a smartphone along with a smartwatch and carry out the activity recognition. Then, we statistically compare the recognition accuracy of four types of physical activities – walking, sitting, standing and driving – with two types of input – data from smartphones, and data from smartphones along with smartwatches. Besides that, we investigate the results of three known classifiers (Decision Tree, Naive Bayes and SVM) and three types of feature vectors as classifier input, based on mean, standard deviation and both combined. The experiment has shown there is significant evidence that the use of data from smartphones combined with smartwatches increases the accuracy in the recognition of the studied activities.

The remainder of this paper is organized as follows. In Section 2, we discuss related work in the area of activity recognition with mobile devices. In Section 3, we outline the experimental design, detailing the research goal, the data collection and preprocessing, the hypotheses, the execution, the results, and the statistical analysis. In Section 4, we present the threats to validity of our work. Finally, in Section 5, we show the conclusions and future work.

II. RELATED WORK

Research in activity recognition can be grouped into distinct phases. In a first moment, authors based their studies only on research devices that collect data through intrusive ways. They use, for example, very heavy devices that could not be used in realistic environments [12], and the devices are designed for a predefined task. The advances in mobile computing emerged a second phase, with works based on mobile devices, such as smartphones [19], and most recently, wearable devices [3]. In this context, data collection is less intrusive and the research achievements seem to be closer to the real world, since proposed solutions can be applied in daily situations. Therefore, some works in the literature focus on mobile activity recognition due to its applicability in many domains [10], [8], [19], [2], [7], [13]. Many of these application domains of activity recognition were summarized by Lockhart et al. [10] that described and categorized a variety of applications based on mobile activity recognition, aiming to direct and encourage other works in the area.

Some works use smartphones for data collection. Among these works, Kwapisz et al. [8] proposed an activity recognition approach based on information collected from smartphone accelerometers. The authors argue that the use of mobile

devices presents a less intrusive way to collect data, different from previous studies that are based on information collected by devices designed only for research purposes. In order to increase the recognition accuracy provided by smartphone data, Kawsar et al. [7] developed a multimodal system that uses data from pressure sensors of shoes along with accelerometer and gyroscope information from smartphones. According to the authors, the solution ensures the data collection even when users are not carrying their mobile devices, and despite using extra information in addition to smartphone data, the solution is wireless, which makes it less intrusive than others previously proposed. Although presenting important results, these works use only smartphones to provide data.

Recently, beside the smartphones, the wearable devices are attracting the attention of the area. Among these devices, stand out the marketable smartwatches. Bieber et al. [2] reported on identified requirements of smartwatch sensors for activity detection (e.g., walking, running, etc.), as well as for inactivity states (e.g., sleeping). The authors introduced a gravity free parameter for the acceleration in the three dimensional space, called Activity Unit. Additionally, they presented an algorithm to distinguish if the user is wearing the device or not and to identify if the user is sleeping. The authors argue that unlike smartphones, the use of smartwatches allows the constant monitoring of physical activities because smartwatches can be used 24 hours a day and some devices are water resistant. Bieber et al. [3] presented challenges and opportunities of smartwatches, as well as their potential applications for assistance and monitoring environments. According to the authors, smartwatches are a good alternative for activity recognition since they are generally used for a longer period than smartphones, most of them are water resistant, additionally, they present good battery life and several sensors that allow non-intrusive monitoring of physical activities. Other smartwatch applications addressed in the paper are gesture recognition (e.g., permanent remote control), sensing (e.g., accelerometer, light, and pressure), and haptic and fast feedback. Although presenting important results, these works use only smartwatches to collect data.

Other works employ smartphones and smartwatches. Among these works, Porzi et al. [13] presented a first prototype of a low-cost system to help visually impaired people, based on the use of a smartphone and a smartwatch. The data from smartwatch sensors is used as input to a gesture recognition algorithm, which runs in the smartphone. The algorithm is based on the Global Alignment Kernel with an SVM classifier. According to the authors, one of the main advantages of the system is that smartwatches can be worn without any prejudice, as they look similar to normal watches. Although this work use both smartphones and smartwatches, each device is used for a different task: the smartwatch for data collection and the smartphone for processing.

As shown in previously mentioned works, the use of sensor data from smartphones in activity recognition is already a reality. However, other devices are emerging as good options for data collection in addition to smartphones, such as smartwatches. But despite its proven applicability presented in some works, they did not experimentally evaluate the impact on the accuracy of the use of these devices combined for activity recognition. Therefore, unlike these studies, we experimentally evaluate the recognition accuracy with distinct scenarios, in

order to investigate whether the combination of data from different devices (i.e., marketable smartwatches and smartphones) can really bring benefits to the area. To accomplish that, we perform human activity identification with different types of feature vectors and different classification methods.

III. EXPERIMENTAL DESIGN

In this section, we present the main results achieved and tasks performed during our work. We specify the research goal, the dataset collection, the instrumentation, as well as specifications about the experiment, and the statistical analysis. Our focus relies on the analysis of a major parameter: accuracy of activity recognition approaches. Our experimental evaluation follows the guidelines presented by Andreas et al. [6].

A. Research Goal

The main goal of this experimental evaluation is to analyze the use of *accelerometer data from smartwatches and smartphones* combined for the purpose of *activity recognition* with respect to its *accuracy* from the point of view of *four physical activities (walking, sitting, standing and driving)* in the context of *pervasive computing*. Therefore, we simultaneously collect accelerometer data from a smartphone along with a smartwatch and carry out the activity recognition using data only from the smartphone and using data from the smartphone and the smartwatch combined.

B. Data Collection Procedure

In order to perform the experimental evaluation, we simultaneously collect sensor data from smartphone and smartwatch accelerometers. To accomplish that, we conduct an experiment in which users performed an average of four activities – walking, sitting, standing and driving¹ – using both devices. To collect the data, we perform the following steps for each participant of the experiment:

- 1) Enter the participant identifier, i.e., first name;
- 2) Place the smartphone in his front right trousers pocket, and the smartwatch in his right wrist (Figure 1);
- 3) Inform which activity will be performed, i.e, walking, sitting, standing or driving;
- 4) Enable the start of the accelerometer data collection by the smartwatch;
- 5) Execute 1 minute of data collection;
- 6) Repeat the steps 3 to 5 for each activity performed.

The participants were mostly undergraduate students aged 20 to 35 years old. At the end of the process, we gather a dataset with the following aspects:

- 13 participants without any physical disability. 12 males and 1 female;
- Information of 4 physical activities.
- 52 log files, each file containing accelerometer data from an activity performed by a participant, and each log containing about 200 lines with:

- 1) Timestamp of the smartphone in milliseconds;
- 2) X-axis value from the smartphone accelerometer;
- 3) Y-axis value from the smartphone accelerometer;
- 4) Z-axis value from the smartphone accelerometer;
- 5) Performed activity;
- 6) Timestamp of the smartwatch in milliseconds;
- 7) X-axis value from the smartwatch accelerometer;
- 8) Y-axis from the smartwatch accelerometer;
- 9) Z-axis from the smartwatch accelerometer.

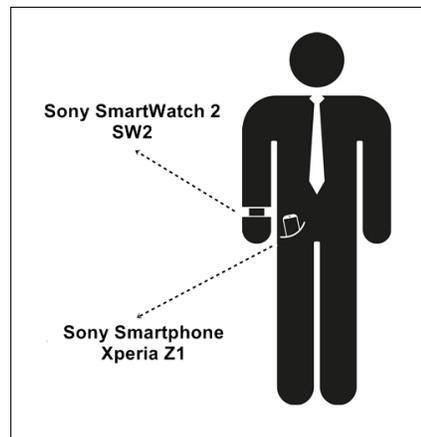


Fig. 1: Illustration of a user with the smartphone in his front right trousers pocket, and the smartwatch in his right wrist.

C. Instrumentation

Among the main tools we use during the experimental evaluation, stand out:

Sony SmartWatch 2 SW2. Android smartwatch that provides integrated accelerometer sensors. We use the SW2 during the data collection phase.

Sony Smartphone Xperia Z1. Android smartphone that provides integrated accelerometer sensors. We use the Z1 during the data collection phase.

Weka². Data mining software in Java. Weka is a collection of machine learning algorithms for data mining tasks, which can either be applied directly to a dataset or called from your own Java code. We use Weka algorithms to classify the activities.

R³. Software environment for statistical computing and graphics. R provides a wide variety of statistics and graphical techniques, and is highly extensible. We use R in the statistical analysis.

D. Data Preprocessing

Before performing the activity classification, we preprocess the raw data collected from the 3-axis accelerometers. This step aims to improve the characterization of the activities and, consequently, to increase the recognition accuracy.

¹We ask the participants to simulate the action of driving, it does not represent the real activity.

²<http://www.cs.waikato.ac.nz/ml/weka/>

³<https://www.r-project.org/>

TABLE I: Examples of feature vector generated from smartphone raw data.

AM X	AM Y	AM Z	STD X	STD Y	STD Z	Physical Activity
0.3052	-9.796	-0.287	1.072955	0.340536	0.479773	Walking
-3.219	-7.428	-6.486	0.021534	0.021778	0.017433	Sitting
1.636	3.624	-10.89	0.937653	0.510336	2.276427	Standing
2.888	-3.562	-9.418	0.246068	0.092064	0.148109	Driving
-0.2014	-9.486	-0.3109	0.675129	1.025669	1.922581	Walking
-3.224	-7.447	-6.47	0.024871	0.050183	0.036129	Sitting

We extract arithmetic mean and standard deviation from the raw data previously collected. To accomplish this task, we calculate both features for every 10 lines of log files. In Table I, it is possible to see some examples of feature vectors generated from the raw data collected from the smartphone, in which “AM X” and “STD X” represent, respectively, the arithmetic mean and the standard deviation based on 10 samples obtained from the x-axis accelerometer.

At the end of the process, we generate six *arff* files that represent the features vectors to be used as input data of classifiers⁴.

E. Experiment Design

In our experiment, we have three independent variables as input source and one dependent variable as output information. The first independent variable is represented by the data source, with the following two levels:

- 1) **Smartphone data:** data collected from a smartphone accelerometer;
- 2) **Smartphone and smartwatch data:** data simultaneously collected from the smartphone and smartwatch accelerometers.

The second independent variable is represented by the classification method, with three levels as follows:

- 1) **Decision Tree:** *J48* classifier from the Weka software tool (default settings);
- 2) **Naive Bayes:** *NaiveBayes* classifier from the Weka software tool (default settings);
- 3) **SVM:** *LibSVM* classifier from the Weka software tool (default settings).

Finally, the third independent variable is represented by the feature vector, with three levels as follows:

- 1) **Arithmetic Mean (AM):** feature vectors obtained by the arithmetic mean calculated from accelerometer raw data;
- 2) **Standard Deviation (STD):** feature vectors obtained by the standard deviation calculated from accelerometer raw data;
- 3) **Standard Deviation and Arithmetic Mean (SA):** feature vectors obtained by the combination of both features.

Our dependent variable is represented by the accuracy of the recognition achieved through a run based on a set of independent variables.

F. Hypotheses

The main research question we want to answer is the following:

P1. Does the use of accelerometer data from smartphones along with smartwatches improve the accuracy of the activity recognition compared to the use of only accelerometer data from smartphones?

To answer that, we formulate the following hypotheses:

H_0 : there is no difference, in terms of accuracy, in the recognition of activities using accelerometer data collected simultaneously from smartphones along with smartwatches and compared to using data only from smartphones, for the classification method i and feature vectors based on j .

H_1 : the use of accelerometer data from smartphones along with smartwatches achieves **greater** accuracy in activity recognition, for the classification method i and feature vectors based on j .

Where i is a classifier of type SVM, Naive Bayes or Decision Tree, and j is a feature vector based on AM, STD or SA. So that, we have 9 null hypotheses.

G. Execution

In order to evaluate the accuracy of classification approaches, in general, data is split into training and test sets, in which the training set comprises a larger portion of data, and it is used to train the approaches, i.e., to provide knowledge to the classification methods about the studied problem. On the other hand, the test set is used to evaluate the accuracy of the approaches [5]. These two datasets must be disjoint, i.e., they should not present elements in common to avoid bias on results.

In our work, we use the 10-fold cross-validation method, which randomly splits the dataset into 10 independent parts, and each part is used once as test set and the remaining as training set. Therefore, we separate the whole data into 90% for training and 10% for testing. Additionally, we repeat the execution three times for each classification method and each feature vector as input data. Thus, we have an amount of 30 result samples per round.

H. Results

In Table II, it is possible to see the average of the accuracies achieved using feature vectors based on standard deviation. For example, the accuracy achieved for *J48* approach (i.e., decision trees) using as input source smartphones only is 66.72%. On the other hand, the same approach with input data

⁴Activity Recognition Repository: <https://goo.gl/Y8NXP1>

from smartphones along with smartwatches obtained greater accuracy of 68.46%.

TABLE II: Average accuracy achieved by 30 runs, using feature vectors based on standard deviation as input data.

Classification methods	Smartphone	Smartphone and smartwatch
J48	66.72%	68.46%
NB	50.73%	59.30%
SVM	56.41%	62.62%

In Table III, it is possible to see the average of the accuracies achieved with feature vectors based on arithmetic mean for the classification methods J48, NB and SVM.

TABLE III: Average accuracy achieved by 30 runs, using feature vectors based on arithmetic mean as input data.

Classification methods	Smartphone	Smartphone and smartwatch
J48	79.40%	78.36%
NB	55.91%	62.05%
SVM	78.30%	81.63%

Finally, in Table IV, it is possible to see the results obtained with feature vectors based on the combination of both features, standard deviation and arithmetic mean.

TABLE IV: Average accuracy achieved by 30 runs, using feature vectors based on standard deviation and arithmetic mean as input data.

Classification methods	Smartphone	Smartphone and smartwatch
J48	87.91%	87.33%
NB	74.59%	80.09%
SVM	89.63%	88.47%

To identify which of the approaches obtained the best accuracy in fact, we perform a pairwise comparison (statistical analysis) for each treatment with one classification method and one feature vector as input data.

I. Statistical Analysis

In statistical analysis some tests are known for two by two comparisons (our case), for example, t-test (parametric) and Wilcoxon test (non-parametric) [4]. Before using t-test, some requirements must be met in both compared samples, data normality and homoscedasticity. Therefore, to choose the proper statistical test, firstly, we verify all the sample profiles, by performing Shapiro-Wilk test to evaluate data normality and Levene's test to evaluate data homoscedasticity. Afterwards, we use t-test in the cases these requirements were met, and Wilcoxon test, otherwise.

We compare the results achieved by three classification methods using data from smartphones only, and data from smartphones and smartwatches combined. Additionally, we investigate three types of feature vectors as input of the classifiers. Because of that, we analyze three hypotheses for each classification method. Besides the smartphone data, as a preliminary result, we also investigated the use of smartwatch data alone. However, it produced lower results, because of that we compare only the smartphone data as well as the combination of both devices.

In Table V, it is possible to see the results of the statistical tests performed during our study to evaluate the 9 proposed hypotheses (Subsection III-F), each test with 95% of confidence (i.e., $\alpha = 0.05$). In order to identify which dataset obtained the greater accuracy, we consider two alternative hypotheses for each null one.

After performing the tests, we have the following results:

- **For J48 classifier:** we accept H_0 for feature vectors based on AM and SA. In the other hand, we reject H_0 and accept H_1 for feature vectors based on STD;
- **For NB classifier:** we reject H_0 and accept H_1 for all types of feature vectors;
- **For SVM classifier:** we accept H_0 for feature vectors based on SA. In the other hand, we reject H_0 and accept H_1 for feature vectors based on AM and STD.

Therefore, we can conclude with 95% of confidence that on 6 of the 9 cases the addition of smartwatch accelerometer data improved the accuracy of activity recognition. Additionally, we do not achieve greater results on any of the 3 remaining situations using only smartphone data as input source.

We still observe that the preprocessing phase used to create the feature vectors directly affects the results. For example, we obtain an accuracy of 56.4% with SVM classifier and feature vectors based on AM (data from smartphones only) and an accuracy of 89.6% with the same classifier, but taking feature vectors based on SA. These results represent a significant improvement in the accuracy of the recognition.

IV. THREATS TO VALIDITY

A. Construct Validity

In this work, we just evaluate the accuracy of activity recognition. However, several evaluation metrics could be applied, for example, precision, recall, insertion, merge, overfill, and confusion matrix [14]. Thus, we suggest in future replications of this experiment to investigate these metrics too.

B. Conclusion Validity

Due to the low number of participants in the dataset collection phase, we have a threat to conclusion validity, since it is recommended to have a larger dataset, for matters of statistical power. Therefore, we intend to continue the dataset collection process with other participants, in order to replicate the experiments with a larger dataset, with different classifiers and different feature vectors.

V. CONCLUSION AND FUTURE WORK

In this paper, we performed an experimental evaluation to investigate the impact, in terms of accuracy, of the activity recognition using accelerometer data from smartwatches along with smartphones as input source.

For the best of our knowledge, this is the first experimental evaluation to measure and statistically analyze the impact of accuracy in activity recognition, achieved by combining simultaneously collected data from marketable wearable devices sensors (i.e. smartwatch accelerometer) and from smartphones. Although some previous works present studies with

TABLE V: Statistical Analysis

Classification methods	Feature vectors	Normal data	Equal variance	p-values
J48	Mean	Yes	Yes	0.7974
	Std	Yes	Yes	0.04658
	Mean and Std	No	N/A	0.9073
NB	Mean	Yes	Yes	7.878e-09
	Std	Yes	Yes	7.927e-12
	Mean and Std	Yes	Yes	0.0002158
SVM	Mean	Yes	Yes	0.009503
	Std	Yes	Yes	1.373e-07
	Mean and Std	Yes	Yes	0.8867

these devices, they did not perform experiments evaluating the accuracy of the recognition with different input feature vectors and different classifiers, as we showed in our work. Furthermore, our work presents significant evidences that the use of marketable smartwatch sensor data in addition to smartphone data can increase the accuracy of activity recognition approaches. Therefore, our study can be used as baseline of future experimental works.

We found some works in literature that focus on features extraction approaches [18], [1]. Therefore, as future work, we will carry out a study about different ways to extract features from accelerometer raw data and replicate this experiment with different input vectors, in order to obtain more representative activity feature vectors, and hence, to get greater accuracy in the activity recognition. Another possible future work is to investigate a greater range of activities based on sensor data from smartwatches and smartphones combined, for example complex activities (e.g., cooking, watching TV, etc) [17]. Additionally, we could do a comprehensive discussion on the sensor data that may affect the physical activity recognition, since mobile devices are equipped with many sensors and provide many kinds of data from them. Then, we could propose new techniques or frameworks to accomplish the activity recognition with these collected data.

ACKNOWLEDGMENT

The authors would like to thank CAPES for supporting this work.

REFERENCES

- [1] A. Akl and S. Valaee. Accelerometer-based gesture recognition via dynamic-time warping, affinity propagation, and compressive sensing. In *Acoustics Speech and Signal Processing (ICASSP), 2010 IEEE International Conference on*, pages 2270–2273, March 2010.
- [2] G. Bieber, M. Haescher, and M. Vahl. Sensor requirements for activity recognition on smart watches. In *Proceedings of the 6th International Conference on PETRA, PETRA '13*, pages 67:1–67:6, New York, NY, USA, 2013. ACM.
- [3] G. Bieber, T. Kirste, and B. Urban. Ambient interaction by smart watches. In *Proceedings of the 5th International Conference on PETRA, PETRA '12*, pages 39:1–39:6, New York, NY, USA, 2012. ACM.
- [4] S. Boslaugh and P. A. Watters. Nonparametric statistics. In *Statistics in a Nutshell: A Desktop Quick Reference (In a Nutshell (O'Reilly))*, pages 207–223. O'Reilly Media, 2008.
- [5] A. Bulling, U. Blanke, and B. Schiele. A tutorial on human activity recognition using body-worn inertial sensors. *ACM Comput. Surv.*, 46(3):33:1–33:33, Jan. 2014.
- [6] A. Jedlitschka and D. Pfahl. Reporting guidelines for controlled experiments in software engineering. In *Empirical Software Engineering, 2005. 2005 International Symposium on*, pages 10 pp.–, Nov 2005.
- [7] F. A. Kawsar, S. I. Ahamed, and R. Love. Smartphone based multimodal activity detection system using plantar pressure sensors. In *Proceedings of the 29th Annual ACM SAC, SAC '14*, pages 468–469, New York, NY, USA, 2014. ACM.
- [8] J. R. Kwapisz, G. M. Weiss, and S. A. Moore. Activity recognition using cell phone accelerometers. *SIGKDD Explor. Newsl.*, 12(2):74–82, Mar. 2011.
- [9] L. Liu, C. Karatas, H. Li, S. Tan, M. Gruteser, J. Yang, Y. Chen, and R. P. Martin. Toward detection of unsafe driving with wearables. In *Proceedings of the 2015 Workshop on WearSys, WearSys '15*, pages 27–32, New York, NY, USA, 2015. ACM.
- [10] J. W. Lockhart, T. Pulickal, and G. M. Weiss. Applications of mobile activity recognition. In *Proceedings of the 2012 ACM Conference on UbiComp, UbiComp '12*, pages 1054–1058, New York, NY, USA, 2012. ACM.
- [11] B. Milosevic and E. Farella. Wearable inertial sensor for jump performance analysis. In *Proceedings of the 2015 Workshop on WearSys, WearSys '15*, pages 15–20, New York, NY, USA, 2015. ACM.
- [12] J. Parkka, M. Ermes, P. Korpipaa, J. Mantjarvi, J. Peltola, and I. Korhonen. Activity classification using realistic data from wearable sensors. *Information Technology in Biomedicine, IEEE Transactions on*, 10(1):119–128, Jan 2006.
- [13] L. Porzi, S. Messelodi, C. M. Modena, and E. Ricci. A smart watch-based gesture recognition system for assisting people with visual impairments. In *Proceedings of the 3rd ACM International Workshop on IMMPD, IMMPD '13*, pages 19–24, New York, NY, USA, 2013. ACM.
- [14] A. Reiss, D. Stricker, and G. Hendeby. Towards robust activity recognition for everyday life: Methods and evaluation. In *Pervasive Computing Technologies for Healthcare (PervasiveHealth), 2013 7th International Conference on*, pages 25–32, May 2013.
- [15] S. Tragopoulou, I. Varlamis, and M. Eirinaki. Classification of movement data concerning user's activity recognition via mobile phones. In *Proceedings of the 4th International Conference on WIMS, WIMS '14*, pages 42:1–42:6, New York, NY, USA, 2014. ACM.
- [16] M. Uddin, A. Salem, I. Nam, and T. Nadeem. Wearable sensing framework for human activity monitoring. In *Proceedings of the 2015 Workshop on WearSys, WearSys '15*, pages 21–26, New York, NY, USA, 2015. ACM.
- [17] Z. Yan, D. Chakraborty, S. Mittal, A. Misra, and K. Aberer. An exploration with online complex activity recognition using cellphone accelerometer. In *Proceedings of the 2013 ACM Conference on UbiComp Adjunct, UbiComp '13 Adjunct*, pages 199–202, New York, NY, USA, 2013. ACM.
- [18] Y. C. Z. F. Yu Liu, Le Wang and Y. Ou. The improvement of behavior recognition accuracy of micro inertial accelerometer by secondary recognition algorithm. *Sensors & Transducers*, 172(0):1 – 6, 2014.
- [19] K. Zhao, J. Du, C. Li, C. Zhang, H. Liu, and C. Xu. Healthy: A diary system based on activity recognition using smartphone. In *MASS, 2013 IEEE 10th International Conference on*, pages 290–294, Oct 2013.

Time Series Classification with Discrete Wavelet Transformed Data: Insights from an Empirical Study

Daoyuan Li Tegawendé F. Bissyandé Jacques Klein Yves Le Traon

University of Luxembourg

{daoyuan.li, tegawende.bissyande, jacques.klein, yves.lettraon}@uni.lu

Abstract

Time series mining has become essential for extracting knowledge from the abundant data that flows out from many application domains. To overcome storage and processing challenges in time series mining, compression techniques are being used. In this paper, we investigate the loss/gain of performance of time series classification approaches when fed with lossy-compressed data. This empirical study is essential for reassuring practitioners, but also for providing more insights on how compression techniques can even be effective in reducing noise in time series data. From a knowledge engineering perspective, we show that time series may be compressed by 90% using discrete wavelet transforms and still achieve remarkable classification accuracy, and that residual details left by popular wavelet compression techniques can sometimes even help achieve higher classification accuracy than the raw time series data, as they better capture essential local features.

1 Introduction

Time series data are commonly found in a variety of real-world applications, including financial data analysis, medical and health monitoring, industrial automation, image recognition, and so forth. Extracting useful knowledge from time series data – a.k.a. time series mining [1] – is now popular but remains a challenging research topic. In particular, time series classification (TSC) attracts significant interest among the researchers and industry practitioners. TSC approaches often implement supervised learning techniques to classify unknown time series instances based on knowledge gained from existing labeled ones.

Due to the abundance and intrinsic high dimensionality of time series data, computing resources such as storage, CPU and memory have become critical bottlenecks in the exploitation of time series. To address these challenges, the research community has proposed efficient dimensionality reduction [2] and representation mechanisms such as SAX [3] and Discrete Wavelet Transforms (DWT). DWT is very popular among both researchers and industry practitioners. In general, wavelets are mathematical functions that process raw data to only keep meaningful oscillations in data values. The earliest wavelet was brought up by Haar

in 1909, but has undergone great development especially since Ingrid Daubechies [4] proved the existence of wavelet families with compact support over an interval [5] and orthogonal translates [6]. Wavelets compression techniques have since then been extensively used in image compression and are part of the JPEG 2000 standard [7]. Wavelet transforms have also been extensively used in medical data analysis including ECG diagnosis [8] and ultrasound image processing [9]. More recently, researchers have applied wavelet transforms in the field of Non-Intrusive Load Monitoring (NILM) [10, 11].

Although wavelets are popular in the research community, the literature lacks a large scale empirical study on the impact of wavelet transformation on the performance of time series mining approaches. In this paper, we seek to investigate this impact on a baseline state-of-the-art time series classification approach. To that end, we compare the classification accuracy of raw uncompressed time series data against transformed data using DWT, including both wavelet approximations and the details, i.e., the residuals or *noises* that are often thrown away during the lossy compression processes. In this way, we are able to separate time series' global features from local defining subsequences.

In this study, we extensively test how discrete wavelet transforms impact TSC accuracy and computational efficiency using 39 openly accessible datasets. Our study suggests that DWT can indeed be useful in time series classification tasks:

- Wavelet transforms can be used to reduce dimensionality of time series data, while at the same time achieving similar classification accuracy compared to using the original uncompressed data. In fact, we demonstrate that time series dimensionality may be reduced by around 90% while still achieving good classification accuracy.
- Wavelets can be used to reduce noises in time series data, so that better classification performance can be achieved after conducting wavelet transform on the original uncompressed data.
- We have further found that, surprisingly, for a few datasets, classification using the compression residual details can be even more effective than using either the original data or the wavelet approximation. This finding suggests that there are specific datasets which are more distinguishable using local features instead of global ones.

The remainder of this paper is organized as follows. We introduce the necessary background information in Section 2 and related work in Section 3. We present experimental details and results in Section 4, before concluding the paper and outlining future work in Section 5.

2 Background

In this section, we present the necessary background to facilitate understanding of this paper. Specifically, we firstly introduce time series and the baseline classification approach used in this paper. Then we show how DWT works and especially how DWT can be applied to time series dimensionality reduction.

2.1 Time Series and TSC

Generally, time series is one important type of temporal data. In the data mining community, time series data are often referred to as ordered lists of numeric values [12]. In this paper, a time series $T = t_0, t_1, \dots, t_{n-1}$, where t_i ($0 \leq i \leq n - 1$) is a finite number and T has a length of n , i.e., $|T| = n$.

Time series classification (TSC) is a common category of tasks that involves learning from existing time series instances (training set) and applying the learned knowledge to assign labels to instances from a testing dataset, where instance classes or labels are often unknown (either this information does not exist or has been intentionally hidden from the classification process). TSC tasks are especially common in application domains such as image and speech recognition (e.g., for recognizing spoken words), medical diagnosis (e.g., for detecting the type of a heart disease in an ECG signal), gesture detection, and so on. Due to its numerous application scenarios, many techniques have been proposed for TSC, including k-Nearest Neighbors, shapelets [13], and bag-of-features [14]. Among them, the Nearest Neighbor (1NN) approach has been proven to work exceptionally well, especially when using Dynamic Time Warping (DTW) [15] for computing the distance metric between a pair of time series samples. Figure 1 shows an example of how DTW works. Unlike Euclidean distance where two time series are aligned point by point, i.e., the i^{th} point in series X is compared against the i^{th} point in another series Y, DTW tries to find the best way to *warp* the time axis and as a result aligns X and Y differently. As shown by the gray dotted lines in Figure 1, an i^{th} point in X can be mapped to a j^{th} point (it is possible that $i \neq j$), or one point in X may even be mapped to multiple points in Y. For Euclidean distance, the gray dotted lines would all be vertical. Thanks to DTW's capability to describe similarities, we consider DTW-based 1NN classification as a reference TSC approach for investigating wavelet transformed time series data.

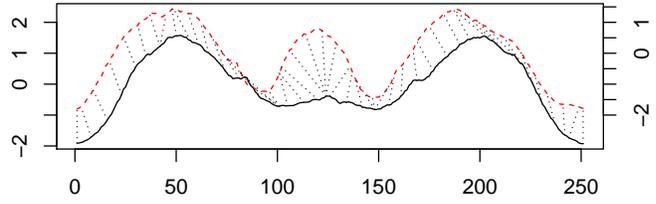


Figure 1: Illustration of how DTW aligns two time series and calculates their distance.

2.2 Discrete Wavelet Transforms

Wavelets are mathematical functions that resemble the shape of wave oscillations, with the constraint for waves to start at 0 and then oscillate to 0 in the end. Mathematically, wavelets are described using two types of functions: the wavelet function (the mother function denoted with $\psi(t)$) and the scaling function (the father function denoted with $\phi(t)$) [16]. These functions have to satisfy the following conditions:

$$\begin{aligned} \|\psi(t)\|^2 &= \int |\psi(t)|^2 dt < \infty, \\ \int |\psi(t)| dt < \infty, \int \psi(t) dt &= 0, \int \phi(t) dt = 1 \end{aligned}$$

The earliest and simplest wavelet function is Haar, whose wavelet function and scaling function are defined as follows:

$$\begin{aligned} \psi_{Haar}(t) &= \begin{cases} -1 & 1/2 \leq t < 1, \\ 1 & 0 \leq t < 1/2, \\ 0 & \text{otherwise.} \end{cases} \\ \phi_{Haar}(t) &= \begin{cases} 1 & 0 < t < 1, \\ 0 & \text{otherwise.} \end{cases} \end{aligned}$$

As illustrated in Figure 2, the Haar wavelet is not a very smooth one, that is, it has a low regularity. Other more sophisticated wavelets – for instance, Daubechies 20 and Symlets 20 – have a higher regularity and thus are able to represent signals more accurately.

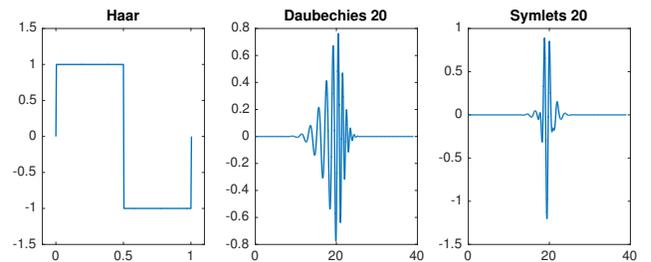


Figure 2: Wavelet functions of Haar, Daubechies 20 and Symlets 20.

In practice, the Haar transform is performed as follows. Given a series $T = t_1, \dots, t_n$, the Haar transform outputs two series: the approximation A and the details D , where for $1 \leq i \leq \frac{n}{2}$,

$$A_i = \frac{t_{2i-1} + t_{2i}}{\sqrt{2}}, D_i = \frac{t_{2i-1} - t_{2i}}{\sqrt{2}}$$

It is clear that the approximations capture the overall *shape* – the global features – of the original series, while the details are the variances – local features – in time series. Figure 3 demonstrates an example of single level, one dimensional Haar transform. As shown, the original signal on the top has been compressed to half of the original size (figure in the middle, note the x-axis label) and the amplitude has been scaled up despite that the overall signal shape is not changed. Note also the grayed area in the signal, it is clear that the approximation becomes smoother after transformation, and that the details shown in the bottom have been dropped.

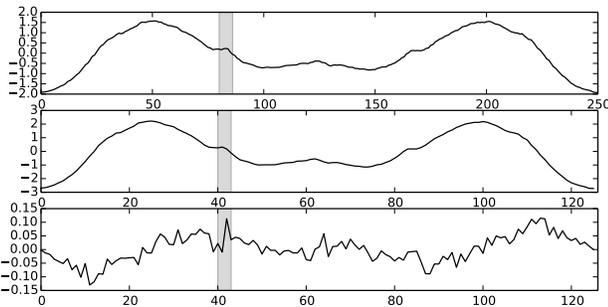


Figure 3: Example of Haar transform: the original signal, the Haar approximation and the residual details.

3 Related Work

Thanks to wavelets’ wide application domains [17], there has been research investigating their performance in various domains. However, to the best of our knowledge, impact of different DWT techniques on TSC has not been done in a generic manner. As a result, in this section we enumerate relevant work to ours in the domains of medical applications, image compression, NILM, and so on.

Addison [8] has conducted a review of both continuous wavelet transform (CWT) and discrete wavelet transform (DWT) on ECG data, concluding that DWT is practically easy to use, while CWT – being more complex and difficult to tune parameters – is able to keep a high resolution in the time-frequency plane, which can result in more accurate identification of components. Another study in wavelet applications in the medical field was done by Pizurica et. al. [9], where the authors reviewed the performance of wavelet denoising specifically in MRI and brain imaging.

To disaggregate electric signals, Duarte et. al. [10] take advantage of CWT in order to extract features from voltage transients and use the extracted features for classification. Gray et. al. [11] use wavelet-based classification for NILM and claim “symlets behave in an identical manner as their

less symmetric Daubechies representation”. But as our empirical study shall demonstrate in the next section, Symlets actually perform better than Daubechies.

Chan et. al. [18] have proposed using Haar for more efficient similarity search, but have not considered other wavelet families. Finally, this work was partly inspired by our previous work, where we have taken advantage of SAX to transform/compress time series data and then build per-class language models [19, 20]. When classifying, we compare time series against models instead of known samples.

4 Experimental Study

In this section, we present our experimental setup and the collected results. In order to facilitate reproducibility, we opt to experiment on publicly available datasets, and further open source our own implementation¹.

Table 1: Characteristics of datasets used in our study.

#	Dataset Name	#Classes	#Training	#Testing	Length
1	ArrowHead	3	36	175	251
2	BeetleFly	2	20	20	512
3	BirdChicken	2	20	20	512
4	Computers	2	250	250	720
5	DistalPhalanxOutlineAgeGroup	3	139	400	80
6	DistalPhalanxOutlineCorrect	2	276	600	80
7	DistalPhalanxTW	6	139	400	80
8	Earthquakes	2	139	322	512
9	ECG5000	5	500	4500	140
10	ElectricDevices	7	8926	7711	96
11	FordA	2	1320	3601	500
12	FordB	2	810	3636	500
13	Ham	2	109	105	431
14	HandOutlines	2	370	1000	2709
15	Herring	2	64	64	512
16	InsectWingbeatSound	11	220	1980	256
17	LargeKitchenAppliances	3	375	375	720
18	Meat	3	60	60	448
19	MiddlePhalanxOutlineAgeGroup	3	154	400	80
20	MiddlePhalanxOutlineCorrect	2	291	600	80
21	MiddlePhalanxTW	6	154	399	80
22	PhalangesOutlinesCorrect	2	1800	858	80
23	Phoneme(readme)	39	214	1896	1024
24	ProximalPhalanxOutlineAgeGroup	3	400	205	80
25	ProximalPhalanxOutlineCorrect	2	600	291	80
26	ProximalPhalanxTW	6	205	400	80
27	RefrigerationDevices	3	375	375	720
28	ScreenType	3	375	375	720
29	ShapeletSim	2	20	180	500
30	ShapesAll	60	600	600	512
31	SmallKitchenAppliances	3	375	375	720
32	Strawberry	2	370	613	235
33	ToeSegmentation1	2	40	228	277
34	ToeSegmentation2	2	36	130	343
35	UWaveGestureLibraryAll	8	896	3582	945
36	Wine	2	57	54	234
37	WordSynonyms	25	267	638	270
38	Worms	5	77	181	900
39	WormsTwoClass	2	77	181	900

4.1 Setup and Datasets

The goal of our study is to investigate how wavelet transformed data may impact TSC performance. Therefore, we do not intend to compare the performance of different classifiers and similarity metrics (which have been empirically studied in [21]). We rely in our study on the most frequently used classification method: Nearest Neighbor Classification (1NN) with DTW distance. In this paper we choose to calculate the classification performance using FastDTW [22],

¹<https://github.com/serval-snt-uni-lu/wavelets-tsc>

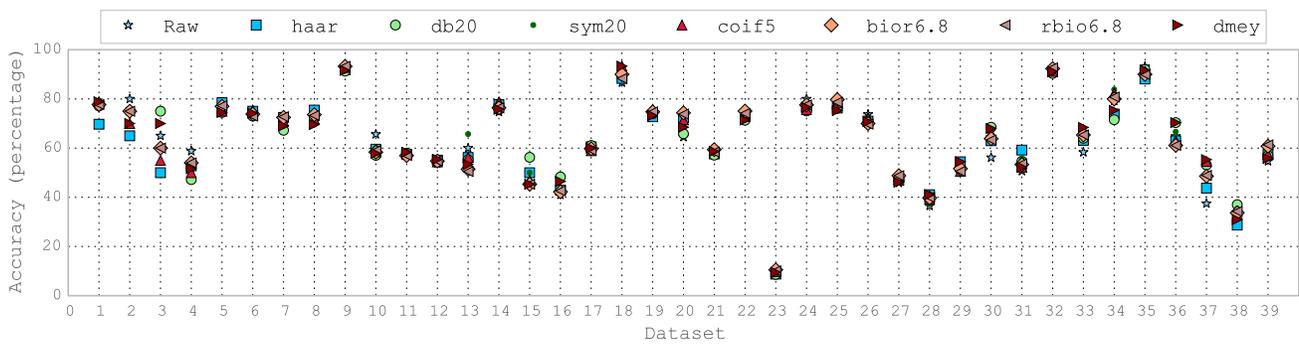


Figure 4: Classification accuracy with FastDTW based 1NN, using original and DWT transformed/compressed data.

which is a DTW approximation with linear time complexity. Although we are more familiar with FastDTW, please note that the UCR Suite [23] is exact instead of approximated and it can be faster than FastDTW.

The datasets that we have experimented on are from the UCR Time Series Classification Archive [24], which contains datasets from various domains ranging from electricity readings and medical signals such as Electrocardiographs (ECGs) to image recognition data. We have specifically chosen to use the `NewlyAddedDatasets`, which contains 39 separate datasets from various domains, with all these datasets sharing a unified file format and internal representation structure, which is convenient for batch processing. The UCR archive provides both the datasets and the ground-truth, i.e., the correct label of each testing instance. Table 1 summarizes the characteristics of these 39 datasets. As shown, this archive comes with predefined training and testing sets, which makes it easier for researchers to compare results in a uniform manner. Besides, it is obvious that several datasets are large in size, for instance, to classify all 7,711 testing instances using 1NN in `ElectricDevices` (#10), there will be $7,711 * 8,926 = 68,828,386$ pairwise comparisons, making the classification process extremely time consuming.

4.2 TSC with Wavelet Transformed Data

As a first step, we seek to investigate how DWT compressed data will impact classification accuracy compared with using raw uncompressed data. Here we transform all the 39 datasets using wavelets from seven well-known families, choosing the single wavelet with the highest regularity from each family. Concretely, the wavelets are: Haar, Daubechies 20, Symlets 20, Coiflets 5, Biorthogonal 6.8, Reverse biorthogonal 6.8 and Discrete Meyer with finite impulse response (FIR) approximation. In this step, all time series – both training instances and testing instances – from each dataset are processed using single level, one dimensional DWT. After transformation, the size of *compressed* data is reduced by half from the original series. Then, we use FastDTW-based 1NN to classify all the testing instances in each dataset. Thanks to the $O(n)$ time complexity of

FastDTW, reducing time series sizes by half means reducing classification time by half. And for DTW with $O(n^2)$ complexity, classification time can be reduced by 75%.

Figure 4 presents the classification accuracy of each wavelet transformed dataset together with that of the original data. Accuracies highlighted in bold indicate equivalent or better performance using wavelet transformed data. We note that, while the compression yields smaller datasets and leads to faster classification, it does not impact the classification accuracy in most cases. Furthermore, in the case of several datasets, the classification accuracy has actually been improved on compressed data. These observations suggest that wavelet transformations are indeed relevant means for noise reduction in TSC tasks.

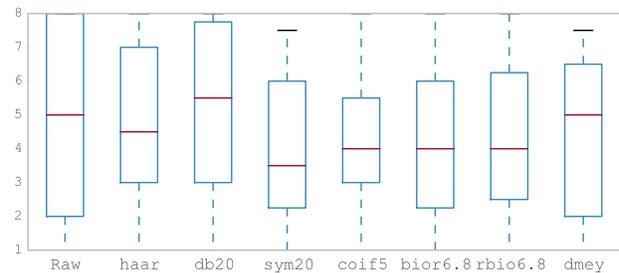


Figure 5: Rank of classification accuracy by approximation of different wavelet transformation.

To further investigate which wavelet family performs better globally, we rank each wavelet family’s classification performance per dataset and draw a boxplot chart of these rankings. As shown in Figure 5, in general wavelet transform data performs better compared with the original data, thanks to wavelets’ noise reduction functionality. Regarding the performance of individual wavelets, **Symlets 20** generally outperforms the rest, including classification using original data. An much to our surprise, **Daubechies 20** in overall performs the worst among all tested wavelets, indicating that the most smooth wavelet may not be the most suitable wavelet for TSC.

4.3 TSC with Residual Details

It is intuitive that when using wavelet transformation, the approximation of original data keeps more relevant infor-

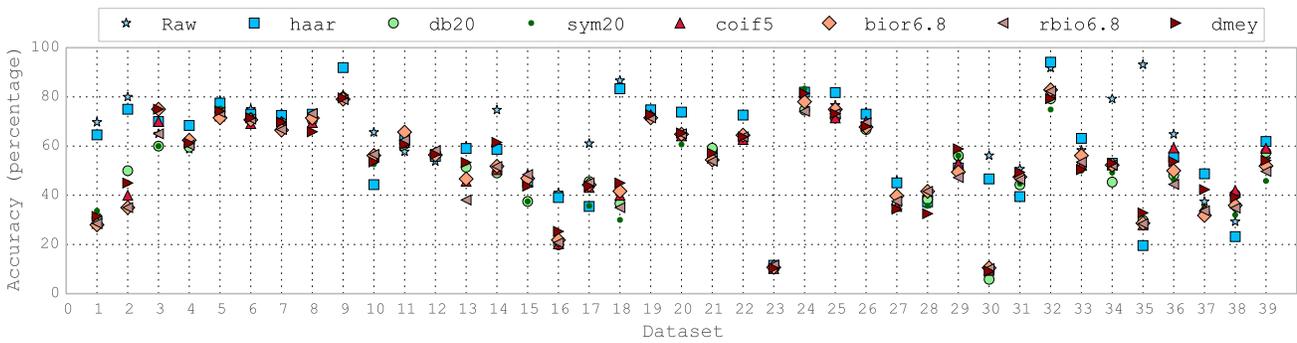


Figure 6: Classification accuracy with FastDTW based 1NN, using original data and residual details from DWT transform.

mation than the residual details. However, in the next experiments, we demonstrate that the residual details may be also useful for TSC and that in some scenarios the *noises* are more discriminative features than the approximations. To prove this seemingly counter-intuitive point, we follow the same procedures as Section 4.2 to compress all datasets, while instead of keeping the approximations, here we drop all of them and consider the residual details, i.e., the noises. Again, to be fair, we compare the classification accuracies using the noises against that using the original untransformed data.

Figure 6 presents the classification accuracies using only the residual details. As shown, although the classification accuracies are generally lower when classifying using only the residuals, they are still surprisingly high in many cases. Especially, for several datasets, classification accuracy using only residuals are similar or higher than using the raw data. Next, we try to rank how discriminative the residuals from each wavelet family are. The boxplot shown in Figure 7 suggests that residuals produced by Haar are most discriminative, being almost as good as the original data. Symlets 20, on the other hand, falls on the other extreme. These two observations are in accordance with the finding in Section 4.2, suggesting that Symlets 20 is best at keeping globally relevant information during transformation. This observation suggests that residuals from DWT compression can also be useful, since these details contain local features that are potentially discriminative.

As a result, we believe that both the approximation and details transformed from the original data are important, due to the fact that the transforms extract two independent discriminative features. While some datasets are more distinguishable using global features, others are more so using local defining features.

4.4 Multi-Level Wavelet Transformation

So far we have only tested the performance of single level wavelet transformation. Since it is possible to conduct wavelet transformation in multiple levels, we seek to investigate how TSC performance can be affected when transforming data through multiple levels. To be specific, we

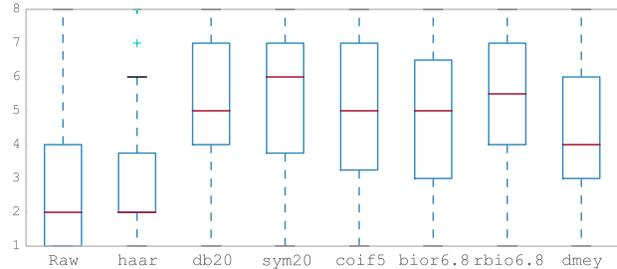


Figure 7: Rank of classification accuracy by residual details from different wavelet transformation.

transform all datasets with Symlets 20 from level 2 till the maximum level. That is, due to different time series lengths, even level 2 transformation is not possible for short time series, while longer time series may be processed using level 6 Symlets 20 decomposition. In Table 2 we present both the percentage of dimension/size reduction (**R**) and the corresponding classification accuracy (**A**) in each level. Note that not all datasets support at least level 2 decomposition due to short lengths, as a result, these datasets are omitted in this table.

As shown in Table 2, although many datasets are compressed by 80% to 90% in size, the classification accuracy using these reduced data can still outperform those using original compressed data. As a result, we think it is safe to claim that multi level wavelet transformation are indeed helpful for TSC tasks when it comes to classifying long time series. Note especially the *HandOutlines* (#14) dataset, due to its extremely high dimensionality, we are able to compress them by up to 97% of the original size, while still received remarkably high classification accuracy. Since FastDWT normally has a time complexity of $O(n)$, this indicates huge time savings in the classification process.

5 Conclusions and Future Work

Discrete Wavelet Transform techniques have matured in the past decades to deliver high data compression rates. Applied to time series data, existing DWT-based lossy compression approaches help to overcome the challenges of storage and computation time. In this paper, we provide assurances to practitioners by empirically showing with var-

Table 2: Classification accuracy with FastDTW based 1NN, using Symlets 20 multi-level decomposition.

#	Raw	Level 2		Level 3		Level 4		Level 5		Level 6	
		R	A	R	A	R	A	R	A	R	A
1	69.7	63.3	78.3								
2	80.0	69.3	65.0	80.9	65.0						
3	65.0	69.3	70.0	80.9	80.0						
4	58.8	71.0	58.4	82.8	55.2	88.8	50.0				
8	71.7	69.3	78.6	80.9	69.9						
11	57.7	69.2	56.5	80.8	59.7						
12	53.8	69.2	55.9	80.8	62.0						
13	60.0	68.2	65.7	79.6	53.3						
14	74.7	73.9	80.0	86.3	79.9	92.4	79.0	95.5	71.1	97.0	70.7
15	45.3	69.3	53.1	80.9	50.0						
16	41.0	63.7	44.1								
17	61.1	71.0	64.0	82.8	63.2	88.8	64.3				
18	86.7	68.5	93.3	79.9	93.3						
23	9.9	72.2	12.3	84.2	12.4	90.2	11.0				
27	45.9	71.0	45.3	82.8	41.9	88.8	39.2				
28	36.5	71.0	36.8	82.8	38.4	88.8	40.0				
29	54.4	69.2	55.0	80.8	51.7						
30	56.2	69.3	72.7	80.9	72.2						
31	50.7	71.0	61.9	82.8	65.9	88.8	67.5				
32	91.8	62.6	91.7								
33	58.3	64.6	68.0								
34	79.2	66.5	81.5	77.6	77.7						
35	93.1	72.0	92.4	83.9	93.2	89.9	91.5				
36	64.8	62.8	55.6								
37	37.5	64.4	61.3								
38	29.3	71.8	35.9	83.8	45.3	89.8	47.0				
39	54.7	71.8	60.2	83.8	64.6	89.8	65.7				

ious datasets and with several DWT approaches that time series classification yields similar accuracy on both compressed (i.e., approximated) and raw time series data. We also show that, in some datasets, wavelets may actually help in reducing noisy variations which deteriorate the performance of mining tasks. In a few cases, we note that the residual details/noises from compression are more useful for recognizing data patterns.

In future work, we plan to extensively investigate the characteristics of time series datasets, in order to empirically correlate successful wavelets techniques per application domains. Dataset characterizations will also help identify which types of time series data can benefit from the use of residual details instead of the approximation data.

References

- [1] T.-C. Fu, "A review on time series data mining," *Engineering Applications of Artificial Intelligence*, vol. 24, no. 1, pp. 164–181, 2011.
- [2] Q. Wang and V. Megalooikonomou, "A dimensionality reduction technique for efficient time series similarity analysis," *Information systems*, vol. 33, no. 1, pp. 115–132, 2008.
- [3] J. Lin, E. Keogh, L. Wei, and S. Lonardi, "Experiencing sax: a novel symbolic representation of time series," *Data Mining and knowledge discovery*, vol. 15, no. 2, pp. 107–144, 2007.
- [4] I. Daubechies, "Orthonormal bases of compactly supported wavelets," *Communications on pure and applied mathematics*, vol. 41, no. 7, pp. 909–996, 1988.
- [5] A. Cohen, I. Daubechies, and P. Vial, "Wavelets on the interval and fast wavelet transforms," *Applied and computational harmonic analysis*, vol. 1, no. 1, pp. 54–81, 1993.
- [6] I. Daubechies, "Orthonormal bases of compactly supported wavelets ii. variations on a theme," *SIAM Journal on Mathematical Analysis*, vol. 24, no. 2, pp. 499–519, 1993.
- [7] D. S. Taubman and M. W. Marcellin, "JPEG2000: Standard for interactive imaging," *Proceedings of the IEEE*, vol. 90, no. 8, pp. 1336–1357, 2002.
- [8] P. S. Addison, "Wavelet transforms and the ECG: a review," *Physiological measurement*, vol. 26, no. 5, p. R155, 2005.
- [9] A. Pizurica, A. M. Wink, E. Vansteenkiste, W. Philips, and B. J. Roerdink, "A review of wavelet denoising in MRI and ultrasound brain imaging," *Current medical imaging reviews*, vol. 2, no. 2, pp. 247–260, 2006.
- [10] C. Duarte, P. Delmar, K. W. Goossen, K. Barner, and E. Gomez-Luna, "Non-intrusive load monitoring based on switching voltage transients and wavelet transforms," in *Future of Instrumentation International Workshop (FIIW), 2012*. IEEE, 2012, pp. 1–4.
- [11] M. Gray and W. Morsi, "Application of wavelet-based classification in non-intrusive load monitoring," in *Electrical and Computer Engineering (CCECE), 2015 IEEE 28th Canadian Conference on*. IEEE, 2015, pp. 41–45.
- [12] T. Rakthanmanon and E. Keogh, "Fast shapelets: A scalable algorithm for discovering time series shapelets," in *Proceedings of the thirteenth SIAM conference on data mining*, 2013.
- [13] L. Ye and E. Keogh, "Time series shapelets: a new primitive for data mining," in *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2009, pp. 947–956.
- [14] M. G. Baydogan, G. Runger, and E. Tuv, "A bag-of-features framework to classify time series," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 11, pp. 2796–2802, 2013.
- [15] G. E. Batista, X. Wang, and E. J. Keogh, "A complexity-invariant distance measure for time series," in *SDM*, vol. 11, 2011, pp. 699–710.
- [16] K. Amolins, Y. Zhang, and P. Dare, "Wavelet based image fusion techniques – An introduction, review and comparison," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 62, no. 4, pp. 249–263, 2007.
- [17] A. K.-m. Leung, F.-t. Chau, and J.-b. Gao, "A review on applications of wavelet transform techniques in chemical analysis: 1989–1997," *Chemometrics and Intelligent Laboratory Systems*, vol. 43, no. 1, pp. 165–184, 1998.
- [18] F. K.-P. Chan, A. W.-c. Fu, and C. Yu, "Haar wavelets for efficient similarity search of time-series: with and without time warping," *Knowledge and Data Engineering, IEEE Transactions on*, vol. 15, no. 3, pp. 686–705, 2003.
- [19] D. Li, T. F. Bissyande, S. Kubler, J. Klein, and Y. Le Traon, "Profiling Household Appliance Electricity Usage with N-Gram Language Modeling," in *The 2016 IEEE International Conference on Industrial Technology (ICIT 2016)*. Taipei: IEEE, 2016, pp. 604–609.
- [20] D. Li, L. Li, T. F. Bissyande, J. Klein, and Y. Le Traon, "DSCo: A Language Modeling Approach for Time Series Classification," in *The 12th International Conference on Machine Learning and Data Mining (MLDM 2016)*. New York: Springer, 2016.
- [21] J. Serrà and J. L. Arcos, "An empirical evaluation of similarity measures for time series classification," *Knowledge-Based Systems*, vol. 67, pp. 305–314, 2014.
- [22] S. Salvador and P. Chan, "Toward accurate dynamic time warping in linear time and space," *Intelligent Data Analysis*, vol. 11, no. 5, pp. 561–580, 2007.
- [23] T. Rakthanmanon, B. Campana, A. Mueen, G. Batista, B. Westover, Q. Zhu, J. Zakaria, and E. Keogh, "Searching and mining trillions of time series subsequences under dynamic time warping," in *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2012, pp. 262–270.
- [24] Y. Chen, E. Keogh, B. Hu, N. Begum, A. Bagnall, A. Mueen, and G. Batista, "The ucr time series classification archive," July 2015, www.cs.ucr.edu/~eamonn/time_series_data/.

An Improved MFD based Regional Traffic Volume Dynamic Control

Yiman Du^{1a}, Jianping Wu^{1b}, Yuhan Jia^{1c}, Ming Xu^{1d}

(1. Department of Civil Engineering, Tsinghua University, Beijing 100084, China)

(a. ymducp@gmail.com; b. jianpingwu@tsinghua.edu.cn; c. yhjiathu@126.com; d. xum@bupt.edu.cn)

Abstract¹: The connection between traffic congestion and regional demand has become a consensus. A reasonable road network inflow control is essential to prevent the occurrence of congestion. With a consideration of deployment of detectors, an improved macroscopic fundamental diagram-based traffic volume dynamic control method via the feedback control is proposed in this study. Based on OD distribution of trips and connectivity of nodes, a novel method is adopted to identify the key nodes, on which the detectors are laid. And Artificial Neural Network is adopted to predict traffic volume of those sections without detectors. As a case study, the proposed methodology is applied to the regional Road network which is located in downtown area of Nanning, China. Adequate survey and analysis are carried out under current road traffic conditions via simulation study. It is proved that the regional traffic volume dynamic control can ensure steady and orderly regional traffic flow, and enhance the mobility in a saturated traffic conditions.

Keywords: regional traffic volume dynamic control; macroscopic fundamental diagram; feedback control; microscopic traffic simulation

1. Background

"Urban area represents the part with densest social-economic and land development activities in the built-up area [1]". Due to the functional characteristics of urban area, an increasing number of cities are facing a sharp contradiction between traffic and development. Massive traffic congestions have become a headache in most cities of China. Whether from the occurrence frequency, intensity, or from the affected area, the overall impact of the traffic congestion on urban development is

increasingly evident.

Typically traffic congestion can be controlled either by increasing road capacity, or reducing traffic demand. From a practical point of view, it's impossible to solve the problem by unlimited expansion of construction of new transport infrastructure. Therefore, it's becoming more important to take better traffic management measures to scientifically regulate traffic flow so as to alleviate traffic congestion and improve urban mobility with the limited resources.

The connection between traffic congestion and regional demand has become a consensus after significant amount of research. When traffic flow reaches the upper limit of the road capacity, congestion occurs. DAGANZO C.F. in 1996 [2] proposed a continuous model of merge operation, which described the process of gradual self-destruction of traffic flow without outside intervention in urban expressway. His study also revealed that the collapse process can be reversed by restricting the ratio of mainline input flow below a critical level. Andre D.P. [3] studied the relation between traffic congestion and traffic equilibrium theory by analyzing traffic congestion during peak periods. The study showed that in commuting fast lane during a peak period, traffic congestion was mainly due to the reach of upper limit of the traffic flow. Therefore, a reasonable road network inflow control is the core to prevent the occurrence of gridlock. For urban areas, traffic volumes control in central region is an effective method to ensure smooth traffic flow.

Traffic volume control means minimizing the number of participants in traffic, shortening the operation time, and reducing road area occupied by participants. When the traffic load in urban road network is close to saturation or saturated, there is no room for the transfer of traffic pressure. Thus traffic volume control has become a good option. Staggered shifts, Congestion charging and vehicle

¹ DOI reference number: 10.18293/SEKE2016-038

license plates restriction rules are the commonly used regional traffic volume control measures. These measures control traffic volume in urban network via policies and regulations. However, the three measures do involve not only traffic problems, but also trade-offs for every participator, and will even affect the public interest then easily lead to disputes. Therefore, transportation managers are more willing to dynamically adjust traffic loads in urban centers from a technical point of view.

A lot of research has been done on signal control. However, isolated or arterial signal control can only partially improve the isolated intersection situation, but cannot cope with large-scale traffic congestion. Many scholars dynamically divided traffic areas into multiple sub-regions for optimal control, and then achieved coordinated control of the entire region [4]. In addition, signal control based on multi-agent technology is also an important area of regional signal control research, and Li Zhenlong[5], Wiering[6], Zhang Hui[7] etc. used modeling and algorithms of coordination of urban traffic signal control based on agent technology to solve the regional coordinated problems and the entire urban traffic network optimization problems. Some scholars also introduced fuzzy logic, neural networks etc. to regional traffic signal control [8]. Existing regional signal control system can improve the overall operating efficiency of road network under unsaturated traffic conditions, but it does not work under the heavy traffic conditions. And at present, there is still no ripe system control program put into use at home and abroad under heavy traffic conditions [9]. A report by the U.S. Federal Highway Administration Commission noted that there was no signal optimization tool which was able to effectively deal with the situation of traffic congestion [10]. Based on the assumption that all links are detector-equipped the fundamental network diagram was exploited by Mehdi etc. to improve mobility in saturated traffic conditions via feedback gating control using microscopic simulation [11]. It is easy to obtain the appropriate information of each link by simulation. However, it is unrealistic to distribute the detectors on every link of road network due to the limitation of funding and complexity of maintenance. Therefore, location and numbers of detectors are important. Based on this, with the consideration of deployment of detectors, an improved

macroscopic fundamental diagram- based traffic volume dynamic control method via the feedback control is proposed in this study. A novel method based on OD distribution of trips and connectivity of nodes is adopted to identify the key nodes, on which the detectors are laid. And Artificial Neural Network are adopted to predict traffic volume of those sections without detectors. As a case study, the proposed methodology is applied to the regional Road network which is located in downtown area of Nanning. Adequate survey and analysis are carried out under current road traffic conditions via simulation study.

2. Methods

2.1. feedback gating

Based on the threshold determined by the Macroscopic Fundamental Diagram (MFD), the strategy of feedback gating control is to control the traffic flow via the gates, which are selected around the boundary of the controlled area. The setting of traffic signals at each gate can be modified accordingly.

According to the MFD, if traffic flow in the controlled area grows beyond the threshold, the exit flow will decrease. Therefore, the inflow should be restricted in order to maximize the throughput of the controlled area. The goal of the controller is to keep the traffic volume of controlled area around the threshold. The logic of feedback control is illustrated in Fig. 1.

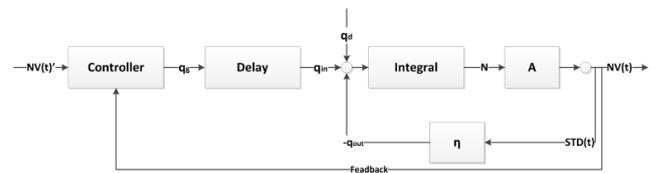


Fig. 1 Regional feedback control logic

$$\dot{N} = q_{in} + q_d - q_{out} \quad (1)$$

$$q_{in} = q_g(t - \tau); \quad (2)$$

Where, q_{in} represents part of traffic flows that entered the controlled area, q_d represents inflows at sections without detectors; q_{out} is the exit flow of controlled area;

2.2. MFD of the study area

Macroscopic Fundamental Diagram (MFD) was first proposed by Godfrey[12] at 1969. The fundamental diagram of traffic flow is a diagram that gives a relation between traffic flow and traffic density. It was found that the exit flow maintains when the inflow grows beyond certain limits based on the simulation results [13]. Geroliminis and Daganzo obtained the MFD of traffic flow and density based on the real data in Japan and simulated data in San Francisco [14], and found that the shape of MFD are affected not by the locations of detectors but by the characteristics of the road network. The curve of the MFD may depend on the traffic demand, however it may be quite stable from day to day [15][16]. Therefore, MFD is the key factor to derive the threshold that maximizes the throughput of the controlled area.

The following equations are adopted to obtain MFD. The Net Volume (NV in veh per h) reflects the horizontal axis of MFD, while the Selected Travelled Distance (STD in veh-km per h) reflects the vertical axis of MFD.

$$NV(k) = \sum_{i \in m} V_i(k) \quad (3)$$

$$STD(k) = \sum_{i \in n} q_i(k) \cdot L_i \quad (4)$$

Where i is the link where detectors are distributed; m is the set of links; $k = 0, 1, 2, \dots$ is the time intervals;

$V_i(k)$ is the estimated traffic volumes in link i during cycle k ; n is the set of links where detectors are distributed; $q_i(k)$ is the measured traffic flow in link i during cycle k ; and L_i is the length of link i . The NV and STD are obtained from the link measurements.

Fig. 2 indicates that a macroscopic fundamental diagram shape (inverse U shape) appears during the simulation period with a quite moderate scatter via different replications. It can be seen in Fig. 2 that the STD value increased with increase of NV during the free-flow period, and STD reached the maximum value in a NV region of 18000 to 19000 veh per h. Then the STD value decreased with the increase of NV beyond the threshold.

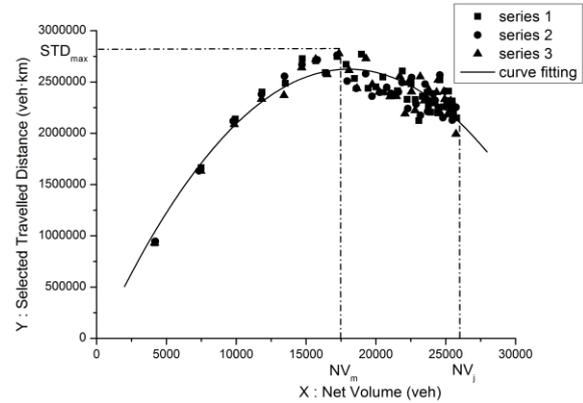


Fig. 2 The MFD of Road Network

2.3. Identification of key nodes

It is easy to obtain the appropriate information of each link by simulation. However, it is unrealistic to distribute the detectors on every link of road network due to the limitation of funding and complexity of maintenance. Therefore, the location and numbers of detectors are important. A better MFD can be obtained if the detectors were distributed on the key links.

Some researchers believe that the relationship between nodes and traffic flow can be described in term of power law distribution [17]. This means that a small number of nodes and sections bear the main road traffic volume. The malfunction of a few key nodes and important sections often lead to a large area of congestion and collapse of the entire road network. Therefore, the detectors are laid on the key nodes and important sections in this study.

A novel method based on OD distribution of trips and connectivity of nodes is adopted in this study [18]. This algorithm extracts statistical information of routes and O-D information of trips from trajectories dataset. Three concepts: trip load, route popularity and node criticality, are introduced in developing a tripartite graph modeled interrelationship among the nodes consists of trip, route and intersection. The relation weight matrix of this tripartite graph can be decided by the proportion of routes and level of intersection. That means, the path which is selected by high load trips with higher probability would be more popular, and the intersection which is passed through by more popular paths would be more critical, and vice versa. Taking relevance between node and O-D distribution into account, which can identify network-wide

key nodes accurately, this algorithm incorporates both topological structure and traffic characteristics. The ratings of trips, paths and intersections can be calculated via interactive process based on the tripartite graph.

2.4. Section traffic prediction with no detector

Due to physical constraints, we can't guarantee that all sections in the area are equipped with detectors, and therefore mathematical statistics analysis is required to obtain the data of sections without detector for further analysis.

As shown in Fig.3, the virtual detector is denoted as X , and the real detectors are denoted as X_1, X_2, \dots, X_6 , which have direct relation with the virtual one. In this study, we consider the volume recorded by virtual detector is only related to the data collected in the vicinity area.

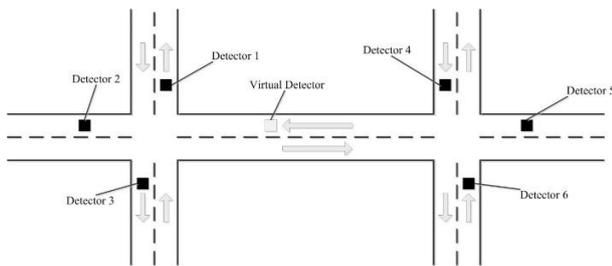


Fig.3 Relationships between virtual and real detectors

Artificial neural networks (ANNs) employ a massive interconnection of simple processing elements that incrementally learn from their environment to capture essential linear and nonlinear trends in complex data, so that it provides reliable predictions for new situations containing even noisy and partial information[19]. ANN (Artificial Neural Network) has a strong adaptability, fault tolerance and self-learning ability, and can fit arbitrarily complex nonlinear models to multidimensional data to any desired accuracy [20]. In order to better predict the traffic flow of those sections without detectors, the BP neural network (Back Propagation Neural Network) was adopted, and massive simulation data were used.

The BPN consists of an input layer, an output layer, and one or more hidden layers. The primary structure of the proposed BP neural model was shown in Figure 4. The number of neurons in the input layer was 7 and the output layer consisted of one neuron. The number of hidden layer was 1. The variables of input neutrons were the signal status of downstream intersection, the green split of

downstream intersection, the signal status of upstream intersection, the green split of upstream intersection, the volume of upstream real detector4, the volume of upstream real detector5, and the volume of upstream real detector6. The output neutrons was the traffic flow of virtual detector.

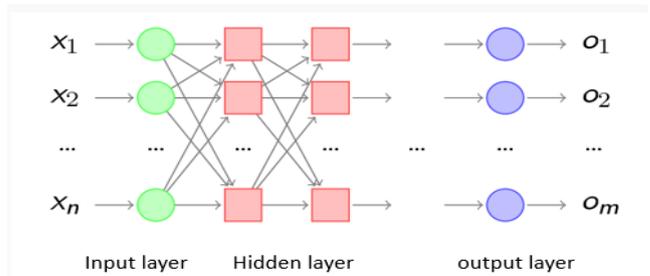


Fig.4 Primary structure of the proposed network

The input data for model training were from the simulation. Multiple simulations were conducted, while the input data were recorded. The best solution was achieved in 1500 epoch. Acceleration was large enough to make the search path oscillate around the minimum indefinitely. The Prediction Model Accuracy was 0.817.

3. Simulation study

3.1. Study area

In this study, the area around the Nanning International Exhibition Center is considered as the controlled region to simulate and demonstrate the Regional Traffic Volume Dynamic Control technology. The China-ASEAN Expo is convened in this area every year, and a large number of state organs and financial institutions are located in this region, which contributes to traffic congestion especially during the rush hour. Therefore, the congestion phenomena are remarkable in this region, which will directly affect traffic conditions of surrounding roads and cause a wide range of traffic jams. 40 intersections are included in this study area, which are shown in figure 6.

3.2. Definition of gates

13 intersections are selected as the controlled "gates" at the boundary of controlled region. The traffic volumes can be restricted at these "gates" and the controlled region can be protected during peak period. The locations of the "gates" are indicated in Fig. 5.



Fig. 5: Demonstration of study area and “gates”

3.3. MFD of the controlled area

Two scenarios are designed to compare the MFD shape between the situations in which the detectors are deployed in each section and the detectors are applied only in key nodes. Multiple replications with different random seeds are carried out for each scenario. The comparison results are shown in Fig. 6. Scenario 1 represents the case that detectors are deployed in each section, and scenario 2 represents the case that detectors are applied only in the key nodes. It is clear that the MFD (inverse-U) shape is indeed occurring in both scenarios. The throughput and efficiency of controlled area increases with the increase of NV before NV reaches NV_m , and then, the throughput degrades when the traffic flow reaches threshold. To maximize the throughput, the NV should be maintained in the optimal range. The NV_m , which are the critical threshold of control, of two sets of data are very close. That means the accurate control threshold can be obtained even though detectors are deployed only in the key nodes. The curve is fitted based on the scatters. It is found that quadratic equation curve provides a good fitting to the scattered point. The coefficient determination is over 85%.

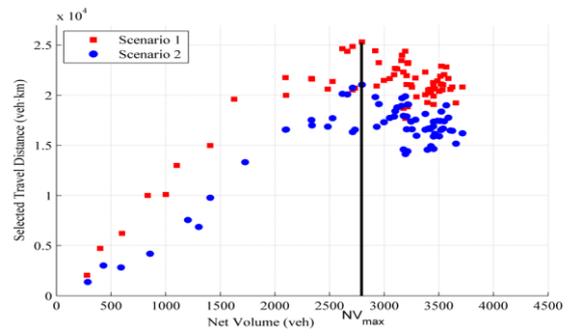


Fig. 6 The comparison of MFD between two scenarios

3.4. Simulation results

Two scenarios are designed to compare the traffic conditions between before and after control. The fixed-time signal control is used in Scenario 1. The gating control logic is implemented in scenario 2, and the threshold of control is set according to the Fig. 6. Because the prediction without detector information would involve additional computation burden, the control interval of the model was set to be 3 minutes. The comparison results are illustrated in Fig. 7 and Table 1, from which the improvements are significant. It is clear in Fig. 7(a) that inflows are controlled to maintain the NV around NV_m when NV reaches the threshold and gating control is switched on around $t=15$ min. And STD stays at high level after the gating control is switched on, compared to scenario 1 without gating control (Fig. 7(b)). Also, the speed of each section shows a significant improvement in Table 1. An average speed increase of 11.96% results for the whole network and some sections even get almost 50% improvements, in contrast to the case without gating control.

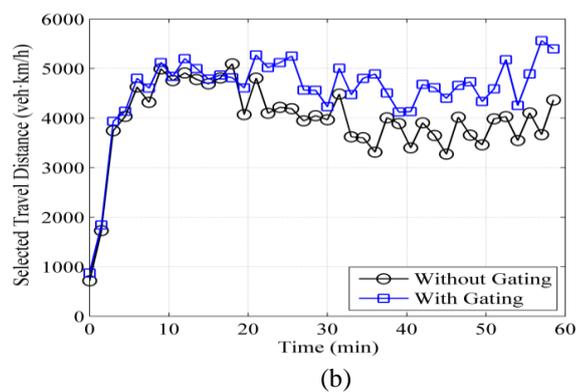
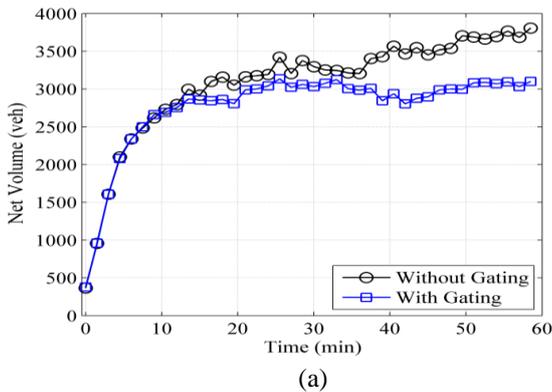


Fig. 7 (a)The comparison of NV between with- and without- gating; (b) The comparison of STD between with- and without- gating

Table 1: Comparison between non-gating and gating cases

Speed/Detector ID	251183	178865	178853	178864	147056	147055	Average
Before control	46.87	7.78	16.47	12.24	9.79	17.58	18.14
After control	49.18	11.59	23.91	14.02	11.06	18.41	20.31
Improve (%)	4.93	48.98	45.17	14.54	12.97	4.72	11.96

4. Conclusion

An improved dynamic regional traffic volume control strategy is proposed based on the feedback gating and MFD concept. The proposed methodology is applied to the area around the Nanning International Exhibition Center. The key nodes are identified based on OD distribution of trips and connectivity of nodes. The microscopic traffic simulation model is employed as the test platform. The results show that the accurate control

threshold can be obtained even detectors are deployed only in the key nodes. The application of feedback control strategy can improve the efficiency and throughput significantly compared to the case without gating control.

5. Acknowledgement

The research is supported by The 863 program with grant number 2012AA063303 and China Postdoctoral Science Foundation with grant number 2013M540102.

[1]T. YANG, J.K. CHEN, H. YU. Research on transportation capacity of city center area. *Urban transport of China*. 2003, 1(1): 13-18.
 [2]C. F. DAGANZO. The nature of freeway gridlock and how to prevent it. *Transportation and traffic theory*. 1996.
 [3]A. de Palma, M. Ben-Akiva, C. Lefevre, and N. Litinsa. Stochastic equilibrium model of peak period traffic congestion. *Transportation Science*, 17(253-274), 1983.
 [4]L.L. ZANG. Optimization of Urban Traffic Signals. Shan Dong University. 2007.
 [5]Z.L. Li, X.H. ZHAO. Coordinated Control of Area Traffic Signal Based on Agent. 32(1): 130-133, 2008.
 [6]M. Wiering, J. Vreeken, J. Van Veenen. Simulation and optimization of traffic in a city. *Intelligent Vehicles Symposium*, 453-458, 2004.
 [7]H. ZHANG, Y.Z. CHEN, Y.Z. YANG. Urban Traffic Coordination Control System Based on Multi-Agent. *Computer and Communications*. 24(2): 94-98, 2006.
 [8]Y. LIN, Z.X. XU, S.B. LI. Simulation Optimization of Traffic Signal Control Parameters. *Journal Of Transportation Systems Engineering and Information Technology*. 10(3): 42-49, 2010.
 [9]Y.X. YU, G.Y. JIANG. Discussion on dynamic dispersal strategy for urban traffic congestion. *Modern Transportation Technology*. 6(4):90-93, 2009.
 [10]R. W. Denney, L. Head and K. Spencer, Signal timing undersaturated conditions, Tech. Rep. FHWA-HOP-09-008 [R], U.S. Department of Transportation, Federal Highway Administration, Washington, DC, USA, 2008.

[11]Mehdi Keyvan Ekbatani, Anastasios Kouvelas, Ioannis Papamichail & Markos Papageorgiou. Congestion Control in Urban Networks via Feedback Gating *Procedia-Social and Behavioural Sciences*. 48:1599-1610, 2012.
 [12]J.W. Godfrey. The mechanism of a road network. *Traffic Engineering and Control*, Vol. 11, 323-327, 1969.
 [13]Y.Y. MA. Research on the signal control systems based on the sub-network coordination. Tongji University. 2010.
 [14]N. Geroliminis and C.F. Daganzo. Existence of urban-scale macroscopic fundamental diagrams: some experimental findings. *Transportation Research Part B*, Vol. 42, pp. 756-770, 2008.
 [15]Y. Ji, W. Daamen, S., Hoogendoorn, S. Laser and X. Qian. Investigating the shape of the macroscopic fundamental diagram using simulation data. *Transportation Research Record*, 2161, pp. 40-48, 2010.
 [16]N. Geroliminis and J. Sun. Properties of a well-defined macroscopic fundamental diagram for urban traffic. *Transportation Research Part B*, Vol. 45, 605-617, 2011.
 [17]J.J. WU, Z.Y. GAO, H.J. SUN..Urban transport system complexity - complex network method and its application. Science Press. 2010.
 [18]M. Xu, J.P. Wu, Y.M. Du. A Method of Key Node Ranking for Road Network Based on Tripartite Graph. *Journal of Beijing University of Posts and Telecommunications*. 37, pp51-54, 2014
 [19]Siame-Irdemoosa, Elnaz, Dindarloo, SR. Prediction of fuel consumption of mining dump trucks: A neural networks approach. *Applied Energy*. 04:77-84, 2015.
 [20]Smith M. Neural networks for statistical modeling. International Thompson Computer Press; 1996.

Traffic Safety Region Estimation Based on SFS-PCA-LSSVM: An Application to Highway Crash Risk Evaluation

Yanfang Yang

School of Traffic and Transportation
Beijing Jiaotong University
Beijing, 100044, China
yangyf@bjtu.edu.cn

Yong Qin

State Key Laboratory of Rail Traffic Control and Safety
Beijing Jiaotong University
Beijing, 100044, China
Corresponding author: yqin@bjtu.edu.cn

Abstract—Accurate real-time crash risk evaluation is essential for making prevention strategy in order to proactively improve traffic safety. Quite a number of models have been developed to evaluate traffic crash risk, by using real-time surveillance data. In this paper, the basic idea of traffic safety region is introduced into highway crash risk evaluation. Traffic safety region aims to describe the safe condition for highway, which means highway is under low risk of crash condition. Sequential forward selection (SFS), principal components analysis (PCA) and least squares support vector machine (LSSVM) are used comprehensively for traffic safety region estimation and classifying the traffic states (safe condition and unsafe condition). The method works by first extracting state variables from the observed traffic variables. Two statistics T^2 and SPE are calculated by SFS-PCA and used as the final state variables for traffic state space. Next, LSSVM is used to estimate the boundary of traffic safety region and identify the traffic states. To demonstrate the advantage of the proposed method, this study develops two crash risk evaluation models, namely SFS-LSSVM model and PCA-LSSVM model, based on crash data and non-crash data collected on freeway I-880N in Alameda. Validation results show that the method is of reasonably high accuracy for identifying traffic states.

Keywords— Traffic safety region, Crash risk evaluation, SFS, PCA, LSSVM

I. INTRODUCTION

Considerable effort has been devoted to shift from reactive (incident detection) to proactive (real-time crash risk prediction) traffic strategies in recent years, as the traffic safety continues to attract growing research interest [1]. Real-time crash risk evaluation models estimate the likelihood of crash occurrence over a short time period, such as 5 min. The results have been utilized in traffic management system. Crash risk evaluation helps discern crash-prone conditions from normal conditions, which is essential for making prevention strategy. For example, the variable speed limit (VSL) system [2] is designed to determine appropriate traveling speed to reduce the crash risk, according to the given current roadway and traffic conditions.

Existing traffic safety analysis can be classified into two categories [3]: aggregate analysis and disaggregate analysis.

Aggregate analysis assumes log-linear relationship [4] or non-linear relationship [5, 6] between various variables and crash frequency, and then tries to estimate the crash frequency. Crash frequency analyses are traditional and feasible, however, more detail about each crash should be known. With the advanced traffic surveillance system, such as loop detector, speed radar, automatic identification system, traffic status prior to crash occurrence could be identified and matched with the corresponding crash. These matched data make disaggregate analysis be possible.

Disaggregate analysis focuses on estimating individual crash risk by using real-time traffic flow data from monitoring detectors. A considerable number of studies have developed regression models to establish a statistical relationship between the crash risk and traffic variables, such as simple/matched-case logistic regression [7, 8], and Bayesian matched-case logistic regression [9]. It is found that traffic flow is significantly related to the risk of crash occurrence. However, a common assumption for regression models is that there is no dependency among the traffic variables, which could cause prediction models to be limited.

In order to overcome the limitation of regression models, data mining/machine learning approaches have been applied to crash risk evaluation. The common used methods include k-nearest neighbor models [10], neural networks [11], Bayesian network models [12, 13] and support vector machines [14-16]. An accompanying issue to the approaches mentioned above is variable selection. Variable selection can help researchers identify and extract meaningful information, which may reduce the prediction running time and improve the prediction results. The widely used variable selection approaches include classification and regression tree [17], random forest [18], frequent pattern tree [19], etc.

Furthermore, some researchers applied reliability analysis of structural engineering to risk evaluation [20]. Reliability analysis is used to distinguish between safe and unsafe conditions by transforming the risk variables into a state space [21], which is the same with the risk evaluation's purpose. This study aims to use machine learning approaches to estimate the safe region of traffic system, termed traffic safety region, and

identify the traffic conditions, i.e. safe state and unsafe state, from the viewpoint of region division. Firstly, state variables are extracted by combining sequential forward selection (SFS) and principal components analysis (PCA) from the observed traffic variables. Statistics T^2 and SPE, calculated from the observed traffic variables, are the state variables for the traffic state space. Then the boundary of the traffic safety region is calculated by two-class least squares support vector machines (LSSVM), and the identification of safe state and unsafe state for the traffic system is performed. The research results will promote a better understanding of the impact of traffic variables on the likelihood of crashes occurrence and the traffic safety region will help transportation professionals monitor the traffic system and develop effective crash prevention strategies on consequential crash events.

II. METHODS

A. Safety Region Theory

Safety region analysis and estimation theory was first applied to safety evaluation for power systems [22]. Zhang et al. [23, 24] applied the idea of safety region into the monitoring and evaluation of the security state for rail system's key equipment. The results found that data mining/machine learning approaches performed well in safety region estimation and safety evaluation.

Supposing $X = \{X_1, X_2, \dots, X_m\}$ is the observed traffic variable vector, m is the number of observed traffic variables. Each X_i is a column vector $[x_{i1}, x_{i2}, \dots, x_{iN}]^T$, $i = 1, 2, \dots, m$, and N is the size of sample. State variables $F = \{f_1, f_2, \dots, f_k\} = g(X)$ are extracted from X , where g is a transform function, and $k \leq m$. For a traffic system, the state space U is constructed by state variables F and consists of two sub regions (traffic safety region E and traffic unsafe region \bar{E}), i.e., $U = E \cup \bar{E}$. Traffic safety region E describes the normal (non-crash) operation area of traffic system. It means, if a state value of traffic system at time t , denoted by P_t , belongs to the traffic safety region E , the traffic system is under the safe condition at time t . Otherwise, the status of traffic system is unsafe (i.e. under high risk of crash condition). Fig. 1 illustrates a two dimensional state space. F_1, F_2 are two state variables and P_1, P_2 are two points which represent safe and unsafe states respectively. The main work of safety region estimation is to obtain the boundary, i.e. a classification and decision-making function to distinguish between safe and unsafe traffic states.

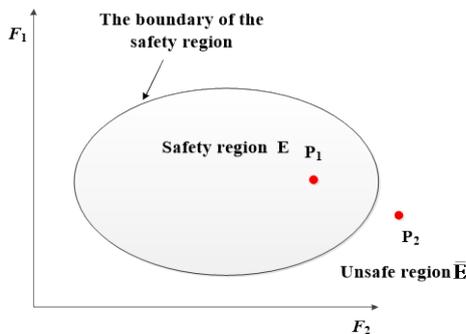


Fig. 1. Schematic diagram of safety region.

B. SFS and PCA

Many factors can contribute to the usefulness of data mining/machine learning algorithm in safety region estimation and crash risk evaluation. The quality of observed traffic variables (e.g. speed, volume, occupancy) is one of these factors. If the observed traffic variables contain irrelevant or redundant features, then the knowledge discovery process during the training becomes noisy and unreliable. In this paper, a state variable extraction method, combining SFS and PCA (SFS-PCA), is considered.

For SFS-PCA method, the input vector V is composed of the observed traffic variable vector X and the corresponding class label vector Y . $Y = \{Y_1, Y_2, \dots, Y_N\}^T$, $Y_l \in \{1, -1\}$, where $l=1, 2, \dots, N$, $Y_l = 1$ corresponds to crash case and $Y_l = -1$ corresponds to non-crash case.

The V could be denoted as:

$$V = \{(X_l, Y_l) | l = 1, 2, \dots, N\} \\ = \{(x_{l1}, x_{l2}, \dots, x_{lm}, Y_l) | l = 1, 2, \dots, N\} \quad (1)$$

where X_l is the l th sample in X , each X_l contains m observed traffic variables.

The goal of SFS-PCA method is to find a minimal set of state variables $F = \{f_1, f_2, \dots, f_k\}$ ($k \leq m$) to represent the observed traffic variables in a lower dimensional state space. The SFS-PCA method could be described as follows:

The best possible subset S of the observed traffic variables is selected by SFS firstly. SFS works in the opposite direction: starting from an empty set, S is iteratively updated by including, in each step, the observe traffic variable vector X_i which results in maximal score $G(S, X, M)$ [25-26]. Thus, the size of S , denoted by d ($d \leq m$), is given by

$$S_d = S_{d-1} \cup \arg \max_{X_i} G(S_{d-1} \cup X_i, X, M) \quad (2)$$

where, M denotes the classification model applied in the task. In this paper, k-nearest neighbor model is used as the classification model.

After the SFS procedure, the final state variable set F is extracted from S by PCA. PCA decomposes S into two subspaces (i.e., a lower dimensional feature subspace composed of principle components and a residual subspace) by multiple projections [27]. Two statistic indicators, T^2 and squared prediction error (SPE), are calculated in the two subspaces respectively. T^2 could reflect the change of the principle component model in feature subspace and SPE could measure the interference and noise in the residual subspace [28].

T^2 and SPE can be calculated using the following equations respectively:

$$T_l^2 = s_l P_b \lambda^{-1} P_b^T s_l^T, \quad l = 1, 2, \dots, N \quad (3)$$

$$SPE_l = s_l (I - P_b P_b^T) s_l^T, \quad l = 1, 2, \dots, N \quad (4)$$

where, s_l is the l th sample in subset S , P_b is the matrix of the b loading vectors, which could be calculated by PCA, I is the identity matrix.

C. LSSVM

LSSVM classifier is one particular sample of support vector machine (SVM). LSSVM classifier maps the input vectors into a high dimensional feature, then finds an optimal separating hyper plane by using maximum Euclidean distance to the nearest point [29, 30].

Given a state variable set $\{(F_l, O_l) \mid l = 1, 2, \dots, N\}$ where F_l is the input data, i.e. the state variables mentioned above, and O_l is the output data, i.e., the classification results. To classify the state variable set, LSSVM has to find the optimal (with maximum margin) separating hyper plane, which could be formulated in the following form:

$$\begin{cases} \min J(\omega, \xi) = \frac{1}{2} \omega^T \omega + \frac{1}{2} \gamma \sum_{l=1}^N \xi_l^2 \\ \text{s.t. } O_l[\omega^T \varphi(F_l) + \eta] = 1 - \xi_l, \quad l = 1, 2, \dots, N \end{cases} \quad (5)$$

where J is an objective function, ω is the normal vector of the separating hyper plane, η is the corresponding bias term and $\varphi(F_l)$ is the nonlinear mapping function, which projects the state variable F_l into a high-dimensional space. γ is the regularization parameter.

The corresponding Lagrange function is:

$$L(\omega, \eta, \xi, \alpha) = J(\omega, \xi) - \sum_{l=1}^N \alpha_l \{O_l[\omega^T \varphi(F_l) + \eta]\} - 1 + \xi_l \quad (6)$$

where α_l is a Lagrange multiplier. According to the conditions for optimality yield, the following equations must be satisfied: $\partial L / \partial \omega = 0$, $\partial L / \partial \eta = 0$, $\partial L / \partial \xi_l = 0$, and $\partial L / \partial \alpha_l = 0$, which could be simplified by $\sum_{l=1}^N \alpha_l \varphi(F_l) = \omega$, $\sum_{l=1}^N \alpha_l = 0$, $\alpha_l = \gamma \xi_l$, and $O_l[\omega^T \varphi(F_l) + \eta] + \xi_l - 1 = 0$ ($l=1, 2, \dots, N$), respectively.

Eliminate ω and ξ , obtain linear equations as follows:

$$\begin{bmatrix} 0 & \mathbf{1}^T \\ \mathbf{1} & \mathbf{K} + \gamma^{-1} \mathbf{I} \end{bmatrix} \begin{bmatrix} \eta \\ \mathbf{a} \end{bmatrix} = \begin{bmatrix} 0 \\ \mathbf{O} \end{bmatrix} \quad (7)$$

where $\mathbf{1}=[1, 1, \dots, 1]^T$, $\mathbf{O}=[O_1, O_2, \dots, O_N]^T$, $\mathbf{a}=[\alpha_1, \alpha_2, \dots, \alpha_N]^T$. \mathbf{I} is an identify matrix and \mathbf{K} is the kernel matrix. With η and \mathbf{a} known, the classification decision-making function is

$$f(F_l) = \text{sgn}[\sum_{l=1}^N \alpha_l K(F_l, F) + \eta] \quad (8)$$

In this study, the radial basis function (RBF) is selected as the kernel function and given as follows:

$$K(F_l, F) = \exp\left(-\frac{\|F - F_l\|^2}{2\sigma^2}\right) \quad (9)$$

where, σ is the width of the RBF kernel.

D. Implementation Procedures for SFS-PCA-LSSVM Method

The implementation procedures for traffic safety region estimation and crash risk evaluation based on SFS-PCA-LSSVM are shown as follows:

Step 1. Collect crash data and non-crash data as the training data for SFS – PCA – LSSVM method. Crash data include crash information (e.g. time, location) and the matched traffic flow data collected from the traffic surveillance system (e.g. speed, volume, occupancy). Non-crash data are traffic flow data in a given time interval when the traffic is under safe condition.

Step 2. Obtain subset S from the observed traffic variable set by using SFS.

Step 3. Process subset S of the observed traffic variables by PCA, and calculate statistics T^2 and SPE. The two statistics form a two-dimensional statistical feature vector for each sample, and the feature vectors would be the final state variable set F .

Step 4. Use the two-dimensional statistical feature vectors as the input data for LSSVM. Classify the traffic states into safe state or unsafe state and obtain the best classified curve which is the boundary of the traffic safety region.

III. DATA PREPARATION

In this paper, crash data and traffic flow data were collected from a 35-mile freeway section from milepost 10.55 to milepost 45.42 on the I-880 freeway in Alameda in the United States. A total of 70 loop detector stations were located along the selected direction (northbound) of the freeway segment. The average spacing between detector stations was approximate 0.5 mile. Crash data and the paired real-time traffic flow data were collected from January 1, 2011 to December 31, 2012. A total of 417 crashes were identified and used for further data analysis.

Traffic data were collected from the highway Performance Measurement System (PeMS) which was computerized database maintained by the California Department of Transportation (Caltrans) [31]. The 30-s raw loop data, i.e. speed, volume and occupancy, for each lane were collected from the Caltrans PeMS database. The raw data further aggregated to 5-min intervals. Each crash was assigned to the nearest loop detector (as shown in Fig. 2), and the traffic data in the time interval between 5 and 10 min prior the crash time were selected to represent the traffic condition [14]. At the same time, the traffic data of upstream and downstream were also extracted. For example, if a crash happened at 13:32, at the milepost 15.46. Traffic condition of nearest loop detector at milepost 15.54 in time intervals 13:20 and 13:25 was the corresponding traffic status for this crash. To eliminate the geometric characteristics' influences on crash risk evaluation [14], matched case-control structure was used to extract non-crash data. For each specific crash case, two non-crash cases, one week before and one week after the crash time, were identified and matched. For example, a crash happened on April 26, 2011, the corresponding non-crash cases (April 19, 2011 and May 3, 2011) at the location of crash occurrence were selected. In this study, a total of 837 non-crash cases

were identified.

For each sample, average and standard deviation values of the speed, occupancy, and volume for the three detectors ($2 \times 3 \times 3 = 18$ variables) constitute the observed traffic variable set for the traffic safety region estimation and crash risk evaluation.

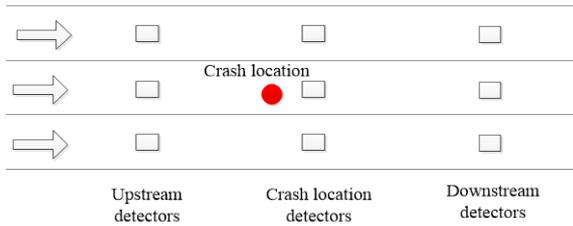


Fig. 2. Illustration of field data collection.

IV. MODELING RESULTS AND DISCUSSION

A. State Variable Extraction Using SFS-PCA

Variables important scores are calculated via SFS and the subset S is determined based on the scores. In this study, 8 observed traffic variables are selected from the 18 observed traffic variables, i.e., downstream standard deviation of speed (DDS), crash location average occupancy (CAO), upstream standard deviation of speed (UDS), crash location standard deviation of occupancy (CDO), downstream average speed (DAS), upstream standard deviation of occupancy (UDO), crash location standard deviation of speed (CDS), and downstream average occupancy (DAO). Furthermore, multicollinearity test for the 8 observed traffic variables has been carried out by using SPSS and the correlation coefficients between two variables in the subset are calculated, as listed in Table 1. The results imply that some of variables exist highly correlated relation, e.g. the correlation between DAO and DAS is 0.825, approximating to 1, which suggests that a further analysis should be conducted on the selected observed traffic variables before being used in the classification model.

TABLE I. CORRELATION MATRIX FOR SELECTED OBSERVED TRAFFIC VARIABLES

	DDS	CAO	UDS	CDO	DAS	UDO	CDS	DAO
DDS	1	0.069	-0.014	0.057	0.171	0	-0.413	0.042
CAO	0.069	1	0.096	-0.357	0.041	-0.202	0.153	-
UDS	-0.014	0.096	1	0.237	0.045	-0.729	-0.422	0
CDO	0.057	-0.357	0.237	1	0.097	-0.312	-0.699	0.121
DAS	0.171	0.041	0.045	0.097	1	0.016	-0.115	0.825
UDO	0	-0.202	-0.729	-0.312	0.016	1	0.283	0.053
CDS	-0.413	0.153	-0.422	-0.699	0.115	0.283	1	-
DAO	0.042	-0.348	0	0.121	0.825	0.053	-0.096	1

In order to eliminate the high correlation among the selected observed traffic variables, the PCA is applied to the observed traffic variable subset. Cumulative percentage of total

variation 80% rule is used to determine the number of components. Finally, three components are chosen. Fig. 3 shows the cumulative proportion for the first 3 components. The corresponding two statistics T^2 and SPE are calculated using (3) and (4), which would be the final input state variable set to the LSSVM method.

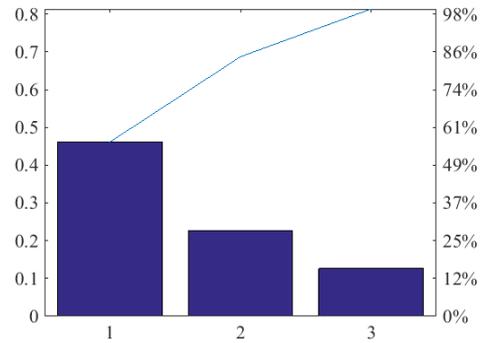


Fig. 3. The cumulative proportion of the first 3 components.

B. Model Results

In this section, the performance evaluation methods used to evaluate the proposed method are presented. The experimental results are given and the observations based on these results are discussed.

The k -fold cross-validation is applied during the classification experiments. The dataset is divided into k subsets, and the holdout method is repeated k times. Each time, one of the k subsets is used as the test set and the other $k-1$ subsets are put together to form the training set. In this study, the 2-fold cross validation is taken, i.e. 50% of the whole data set is used as the training dataset and the rest 50% of the whole data is used as the validation dataset. The classification result is shown in Fig. 4. The left column is the classification result of training dataset, and the traffic safety region boundary is also estimated. The right column plots the testing data points on the classified region. In Fig. 4, it is intuitive that the SFS-PCA-LSSVM method works efficiently in classifying the traffic statuses. However it is necessary to evaluate these results in a quantitative way.

The classification accuracy for the dataset is measured by correct rate (CR) according to

$$CR = \frac{\text{the total number of samples correctly classified}}{\text{the total number of samples}} \times 100\% \quad (10)$$

TABLE II. CR FOR THREE METHODS IN DIFFERENT TRAINING DATASET

Training dataset	SFS-LSSVM	PCA-LSSVM	SFS-PCA-LSSVM
2-fold	75.54%	67.70%	88.16%
3-fold	76.12%	68.34%	88.28%
4-fold	77.19%	68.82%	88.84%

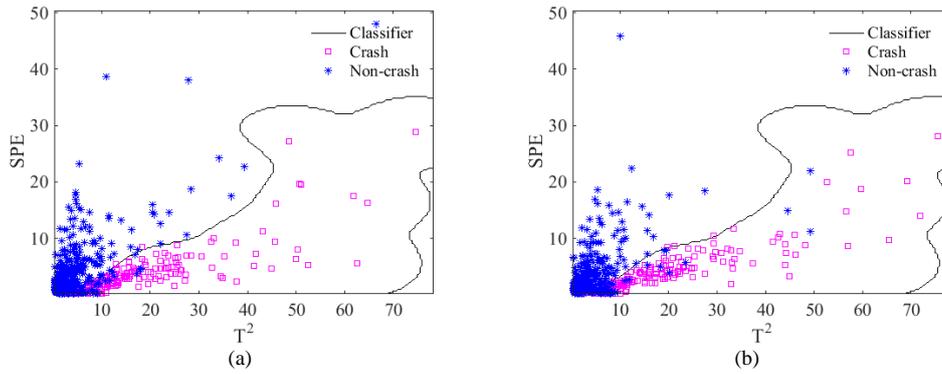


Fig. 4. 2-Fold classification results: (a) training dataset (b) test dataset.

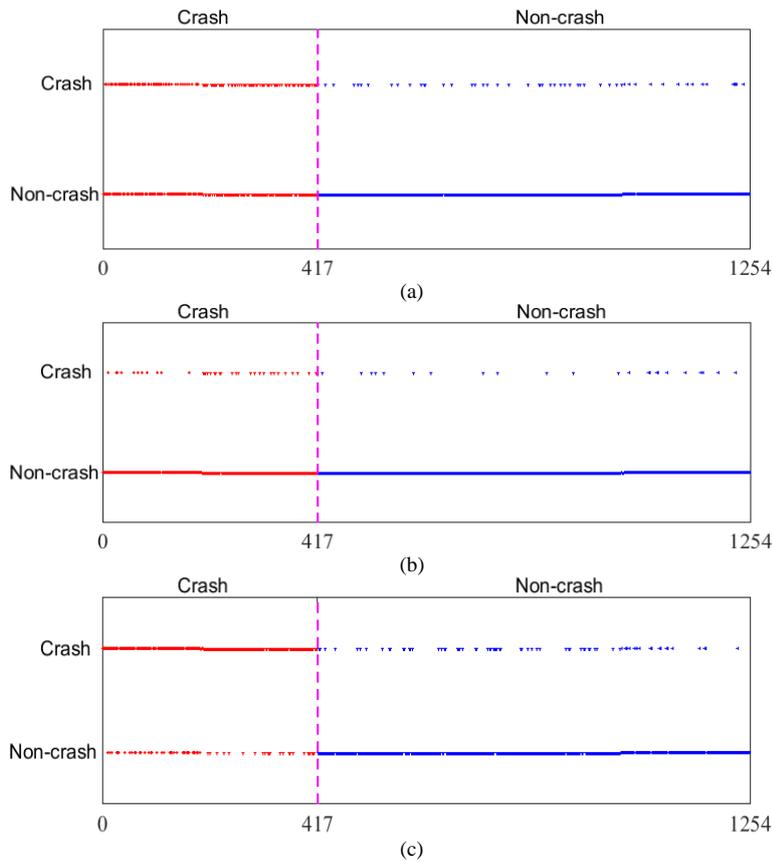


Fig. 5. The classification results of three methods: (a) SFS-LSSVM (b) PCA-LSSVM (c) SFS-PCA-LSSVM

The classification performance of the proposed method was compared with the SFS-LSSVM and PCA-LSSVM methods as listed in the 1st row of the Table 2. Furthermore, in order to demonstrate the classification performance, the classification results of three methods are plotted. As shown in Fig. 5, the horizontal axis represents the number of state points. The 1st~417th points are crash state samples, the 418th ~ 1254th are non-crash state points. The whole area is divided into two subareas by pink dotted lines. Vertical axis from bottom to top represents the two classes of data

samples. It can conclude that the proposed method performs better than other two methods.

The traffic safety region estimation is dependent on the size of training dataset. In this study, two additional cross-validation experiments are conducted, i.e. 3-fold cross-validation and 4-fold cross-validation. The CR values for the three models mentioned above are calculated. As listed in Table 2, all the CR values of SFS-PCA-LSSVM are higher than that of other two models. While the size of training dataset becomes bigger, the corresponding CR values increase. It can be conclude that for different size of

training dataset, sufficient training data may improve the classification results.

V. CONCLUSIONS

This paper presents an intelligent method based on SFS-PCA-LSSVM for traffic safety region estimation and traffic state classification. The traffic condition data, including crash data and non-crash data, is obtained and the dimension of these traffic variables is 18. SFS method is used and selects 8 observed variables as the sub variable set, which could be found that some of observed variables exist high correlation. In order to reduce the high correlation among selected variables, the PCA is applied to extract state variables from the variable subset. Two statistics, i.e. T^2 and SPE, are calculated and given to LSSVM classifier in classification stage of SFS-PCA-LSSVM traffic safety region estimation system. Final experimental results show that the classification accuracy rates of the proposed method are all reasonable high accuracy, i.e. all above 88%, and SFS-PCA-LSSVM method outperforms two given methods, SFS-LSSVM and PCA-LSSVM methods.

ACKNOWLEDGMENT

This research was supported by the National Science & Technology Support Program of China (Grant No. 2014BA G01B02), and Beijing Municipal Institute of Science and Technology (Grant No. PXM2015_178215_000008).

REFERENCES

- [1]. C. Shew, A. Pande, and C. Nuworsoo. "Transferability and robustness of real-time freeway crash risk assessment". *Journal of Safety Research*, Vol. 46 , pp. 83-90, 2013
- [2]. S. Pan, L. Jia, N. Zou, and S. Park. "Design and implement of a variable speed limit system with travel time display – a case study in Maryland," USA. In 17th ITS World Congress, Busan 2010
- [3]. Q. Shi, and M. Abdel-Aty. "Big data application in real-time traffic operation and safety monitoring and improvement on urban expressways," *Transportation Research Part C: Emerging Technologies*. 2015 in press
- [4]. F. Guo, X. Wang, and M. Abdel-Aty. "Modeling signalized intersection safety with corridor-level spatial correlation," *Accident Analysis and Prevention*, Vol.42 , pp. 84-92, 2010
- [5]. R. Yu, and M. Abdel-Aty. "Investigation the different characteristics of weekday and weekend crashes," *Journal of Safety Research*, Vol. 46, pp. 91-97, 2013
- [6]. C. Wang, M. A. Quddus, and S. G. Ison. "Impact of traffic congestion on road accidents: a spatial analysis of the M25 motorway in England," *Accident Analysis and Prevention*, Vol. 41, No.4, pp.798-808, 2009
- [7]. M. Abdel-Aty, N. Uddin, F. Abdala, A. Pande, and L. Hsia. "Predicting freeway crashes based on loop detector data using mathed case-control logistic regression," *Transportation Research Record: Journal of Transportation Research Board*, No.1897, Transportation Research Board of the National Academies, Washington, D.C. , pp. 88-95, 2004
- [8]. Z. Zheng, S. Ahna, and C. "Monsere. Impact of traffic oscillations on freeway crash occurrences," *Accident Analysis and Prevention*, Vol. 42 , pp. 626-636, 2010
- [9]. M. Abdel-Aty, H. M. Hssan, M. Ahmed, and A. S. Al-Ghamdi. "Real-time prediction of visibility related crashes," *Transportation Research Part C: Emerging Technologies*, Vol. 24, pp.288-298, 2012
- [10]. Y. Lv, S. Tang, and H. Zhao. "Real-time highway traffic accident prediction based on the k -nearest neighbor method," *International Conference on Measuring Technology and Mechatronics*

- Automation, Vol. 3, pp. 547-550, 2009
- [11]. M. Abdel-Aty, A. Pande, A. Das, and W. J. Knibbe. "Assessing safety on Dutch freeway with data from infrastructure-based intelligent transportation systems," *Transportation Research Record: Journal of Transportation Research Board*, No. 2083, Transportation Research Board of the National Academies, Washington, D.C., pp. 153-161, 2008
- [12]. O. H. Kwon, W. Rhee, and Y. Yoon. "Application of classification algorithm for analysis of road safety risk factor dependencies," *Accident Analysis and Prevention*, Vol. 75, pp. 1-15, 2015
- [13]. C. Xu, P. Liu, W. Wang, and Z. Li. "Identification of freeway crash-prone traffic conditions for traffic flow at different levels of service," *Transportation Research Part A: Policy and Practice*, Vol. 69 , pp. 58-70, 2014
- [14]. R. Yu, and M. Abdel-Aty. "Utilizing support vector machine in real-time crash risk evaluation," *Accident Analysis and Prevention*, Vol.51, pp. 252-259, 2013
- [15]. R. Yu, and M. Abdel-Aty. "Analyzing crash injury severity for mountainous freeway incorporating real-time traffic and weather data," *Safety Science*, Vol. 63, pp. 50-56, 2014
- [16]. R. Gang, and Z. Zhou. "Traffic safety forecasting method by particle swarm optimization and support vector machine," *Expert Systems with Application*, Vol.38, pp.10420-10424, 2011
- [17]. M. Hosain, and Y. Muromachi. "Understanding crash mechanism on urban expressways using high-resolution traffic data," *Accident Analysis and Prevention*, Vol.57, pp. 17-29, 2013
- [18]. M. M. Ahmed, and M. Abdel-Aty. "The viability of using automatic vehicle identification data for real-time crash prediction," *IEEE. Transactions on Intelligent Transportation System*, Vol. 13, pp.459-468, 2012
- [19]. L. Lin, Q. Wang, and A. W. Sadek. "A novel variable selection method based on frequent pattern tree for real-time traffic accident risk prediction," *Transportation Research Part C: : Emerging Technologies*, Vol.55, pp. 444-459, 2015
- [20]. R. Yu, Q. Shi, and M. Abdel-Aty. "Feasibility of incorporating reliability analysis in traffic safety investigation," *Transportation Research Record: Journal of Transportation Research Board*, No. 2386, Transportation Research Board of the National Academies, Washington, D.C., pp. 35-41, 2013
- [21]. A. S. Nowak , and K.R. Collins. *Reliability of Structures*. CRC Press, 2012
- [22]. F. Wu, and S. Kumagai. "Steady-state security regions of power systems," *IEEE Transaction on Circuits and Systems*, Vol. 29, pp. 703-711, 1982
- [23]. Y. Zhang, Y. Qin, Z. Xing, L. Jia, and X. Cheng. "Roller bearing safety region estimation and state identification based on LMD-PCA-LSSVM," *Measurement*, Vol. 46, pp. 1315-132, 2013
- [24]. Y. Zhang, Y. Qin, Z. Xing, L. Jia, and X. Cheng. "Safety region estimation and state identification of rolling bearing based on statistical feature extraction," *Shock and Vibration*, Vol. 20, pp.833-846, 2013
- [25]. Y. Liu, and Y. F. Zheng. "FS_SFS: A novel feature selection method for support vector machine," *Pattern Recognition*, Vol.39, pp.1333-145, 2006
- [26]. J. Pohjalainen, O. Räsänen, and S. Kadioglu. "Feature selection methods and their combinations in high-dimensional classification of speaker likability, intelligibility and personality traits," *Computer Speech and Language*, Vol.29, pp.145-171, 2015
- [27]. K. Pearson. "On lines and planes of closest fit to systems in space," *Philosophical Magazine*, Vol. 2, pp. 559-573, 1901
- [28]. A. J.Richard and W. W. Dean. *Applied multivariate statistical analysis*. Six ed., Prentice Hall, New Jersey, 2007
- [29]. D. Çalışır, and E. Dogantekin. "A new intelligent hepatitis diagnosis system: PCA- LSSVM," *Expert Systems with Application*, Vol. 38, pp.0705-10708, 2011
- [30]. H. Xu, and G. Chen. "An intelligent fault identification method of rolling bearing based on LSSVM optimized by improved PSO," *Mechanical Systems and Signal Processing*, Vol.35, pp. 167-175, 2013
- [31]. C. Xu, P. Liu, W., Wang, and Z. Li. "Evaluation of the impacts of traffic states on crash risk on freeways," *Accident Analysis and Prevention*, Vol. 47 , pp. 162-171, 2012

Sensor-Based Detection Approach for Passenger Flow Safety in Chinese High-speed Railway Transport Hub

Xie Zhengyu

School of Traffic and Transportation
Beijing Jiaotong University
Beijing, China
xiezh2001@gmail.com

Qin Yong

State Key Laboratory of Rail Traffic Control and Safety
Beijing Jiaotong University
Beijing, China
qingyong2146@126.com

Abstract—Passenger flow safety detection in high-speed railway transport hub is considered in this paper. For accurately detecting the passenger flow safety, an improved watershed algorithm and a recognition algorithm are respectively proposed in sensor-based detection process. Computational experiments on sensor data from a specific Chinese high-speed railway transport hub show that the proposed algorithms are effective and efficient for detecting the passenger flow safety in high-speed railway transport hub.

Keywords—intelligent transportation system; sensor-based detection; passenger flow safety; high-speed railway transport hub

I. INTRODUCTION

Over the past seven-year period, a huge high-speed railway network has been built in China, with lines in operation amounting to 11,132 km. The rapid developing of high-speed railway leads the passenger distribution quantity increasing sharply in transport hubs, which make the safety management of hub confront with serious challenges. Current video surveillance systems in high-speed railway transport hubs are relatively simple, which cannot automatically detect the potential safety hazards in high density passenger flow, only can passively respond for the emergency. According to the stringent safety detection demands, the detection method of video surveillance systems needs become more intelligence and automatic.

Intelligent video surveillance is an important content in computer vision study, which uses computer vision technology and artificial intelligence technology to analyze, process and describe the image sequences of video surveillance. The powerful processing ability of computer can rapidly filter out interferential information, extract objective information, and achieve the detecting, locating and tracking of surveillance objectives. Currently, the intelligent video surveillance is widely applied in transportation fields including: moving people and vehicles detecting and tracking [1-3], abnormal behavior detecting [4-6], vehicle shape, color and plate number recognition [7,8], sensitive area intrusion detection [9,10],

unusual crowd detection [11,12], etc. With more video surveillance sensors used in transport hubs, the studies focus on the moving object detection in hubs become a new hotspot. However, in surveillance scenes, the interference of weather, illumination, shadow and repetitive motion of passengers make video surveillance difficult to detect objects fast and reliably.

The passenger flow safety detection in high-speed railway transport hub belongs to the moving object detection problem, which has been investigated in many literatures. The universal process of moving object detection, and all typical algorithms and their advantages were expounded, then summarized algorithms characteristics, and compared the performances of some algorithms. Finally, pointed out key issues and directions of future study in this area [13]. According to comparing moving object detection methods and typical background modeling methods, a suitable method was selected for background modeling in high-speed railway transport hub [14]. An efficient adaptive segmentation algorithm was developed for color video surveillance sequence in real time with non-stationary background [15]. A hybrid temporal-spatio forecasting approach was proposed to obtain the passenger flow safety in high-speed railway transport hub. The approach combined temporal forecasting based on radial basis function neural network and spatio forecasting based on spatial correlation degree [16].

According to the literature review above, several moving object detection approaches were present. However, there is no universal method can solve different problems. Detection method must be developed based on the detection objects and applied places. For passenger flow detection, a background model based on Dempster-Shafer theory and a recognition algorithm based on connected domain were developed [18]. In this paper, based on the detection method developed in Ref [18], we add an image segmentation algorithm in sensor-based detection process and propose an improved recognition algorithm to improve the accuracy of passenger flow safety detection in high-speed railway transport hub.

The rest of paper is organized as follows. Section 2 introduces the process of passenger flow safety detection based on surveillance sensors. Section 3 proposes an improved watershed algorithm based on tag obtained automatically. A recognition algorithm based on connected domain characters is developed in section 4. Section 5 takes experiments on sensor data from a specific Chinese high-speed railway transport hub to verify the algorithms we proposed. Section 6 concludes the paper and provides the direction for future work.

II. PASSENGER FLOW SAFETY DETECTION PROCESS BASED ON SURVEILLANCE SENSORS

According to the basic process of moving object detection, the passenger flow safety detection process based on surveillance sensors in high-speed railway transport hub is designed as follows.

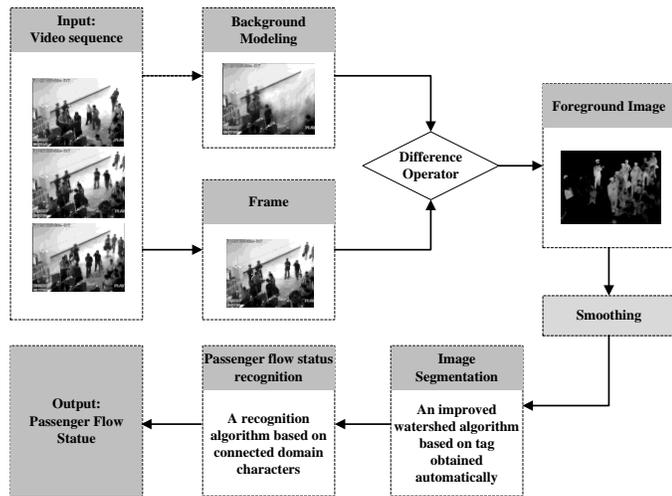


Figure 1. Passenger flow safety detection process based on surveillance sensors.

(1) Input and output: the process input is the video sequence obtained from video surveillance sensors in high-speed railway transport hub, and the output is the passenger flow status of detection areas.

(2) Background modeling and updating: according to the characteristics of detection areas in high-speed railway transport hub, a background model is developed to build a reliable background image, and an updating strategy is proposed to update the background image for adapting the dynamic circumstance change in hub.

(3) Image segmentation: based on the characteristics of passenger flow foreground images, an image segmentation algorithm is proposed to accurately filter out interferential information and extract object information.

(4) Passenger flow safety recognition: a passenger flow safety recognition method is proposed to recognize the passenger flow safety from the extract object information which is obtained by image segmentation.

In the process mentioned in Fig.1, the background modeling and updating have been studied [17]. In this paper, we consider the image segmentation algorithm and passenger

flow status recognition method. Firstly, an improved watershed algorithm based on tag obtained automatically is proposed to avoid excessive segmentation and reduce useful information losing. Then, for solving severe overlap in passenger flow image, a recognition algorithm based on connected domain characters is developed. The two algorithms are detailed in the following sections.

III. AN IMPROVED WATERSHED ALGORITHM BASED ON TAG OBTAINED AUTOMATICALLY

In order to meet high segmentation accuracy demands of high-speed railway transport hub, the image segmentation algorithm need to reduce useful information losing, and extract the passenger flow information in the segmentation process. As a classical image segmentation method, the watershed algorithm has the advantages of edge location accuracy, simple operation, etc, and has been applied in many fields. However, traditional watershed algorithm has disadvantage of excessive segmentation, which seriously interferes the recognition of object edge. For avoiding the excessive segmentation, several improved algorithms are proposed. The improved watershed algorithm based on tag is one of improved algorithms, improved from the segmentation theory of watershed algorithm, and essentially solve the excessive segmentation. So we select this algorithm as segmentation algorithm.

In typical tag extracting method, threshold is determined by subjectively manual setting, which is suitable for small quantity image processing. But the setting method cannot meet the timeliness requirement of video surveillance in high-speed railway transport hub, and has low adaptation for the dynamic circumstance change. So according to the demands of segmentation in high-speed railway transport hub, we propose an improved watershed algorithm based on tag obtained automatically. The algorithm firstly sorts the gradient of each pixel in the image, tags the threshold by using the image gray information, sets the tag value as the minimum image gradient, and modifies the gradient image. Then, use the immersion to process the modified gradient image. The algorithm process is shown in Fig.2.

Main steps of the algorithm are described as follows.

Step 1: calculate the gradient of each pixel in the image, and scan the whole image to get the probability density P_i of each gradient. The position of each pixel in sorted array is determined by the calculation of graded distribution cumulative probability and the gradient of each pixel. In the sorted array, the pixel with the low gradient value is put in the front of array.

Step 2: process the pixels in accordance with the sequence from low to high. The pixels having same gradient are processed as one gradient hierarchy.

Step 3: count the maximum gradient set U_{max} and minimum gradient set U_{min} , calculate the average gradient h_{max} of the maximum gradient set and the average gradient h_{min} of the minimum gradient set. Then count the secondary minimum gradient set U_{min}' from all pixels whose gradient is greater than h_{min} , and calculate the average gradient h_{min}' of the secondary

minimum gradient set. Based on the data mentioned above, the threshold T is calculated as follows:

$$T = h_{\min} + h'_{\min} - \sigma \quad (1)$$

Where σ is the mean square error of U_{\max} .

Step 4: according to the threshold obtained in step 3, tag the minimum gradient, and sort the pixels whose gradients are lower than T into one gradient hierarchy.

Step 5: process the h gradient hierarchy, putting all pixels whose neighborhoods have been tagged into a first-in first-out (FIFO) queue.

Step 6: if the queue is empty, go to step 8. Otherwise, go to step 7.

Step 7: process the neighborhoods of the first pixels in the queue. If the neighborhood of pixel has been tagged, updates the pixel tag based on the neighborhood tag, and then delete this pixel from the queue, go to step 6. If the neighborhood of pixel has not been tagged, put this neighborhood into the FIFO queue, update the queue, go to step 6.

Step 8: scan the h gradient hierarchy to check if any pixel is not tagged. If there is a pixel not tagged, add 1 to the current tag value, and take this value as the tag value of the untagged pixel, go to step 4. Otherwise, go to step 9.

Step 9: set $h = h + 1$, if $h \leq \max_hierarchy$, then go to step 4. Otherwise, segmentation finishes.

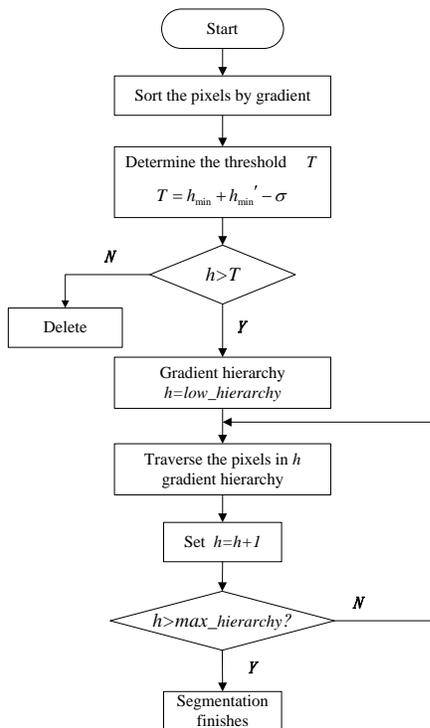


Figure 2. Improved watershed algorithm process

IV. A RECOGNITION ALGORITHM BASED ON CONNECTED DOMAIN CHARACTERS

In ideal conditions, the passenger objects should be segmented in different connected domains based on the previous processing. According to the amount of the connected domains, the passenger quantity can be calculated, and the other passenger status could be obtained by passenger quantity. But in practical application, the crowd in passenger flow in high-speed railway transport hub makes passenger objects overlapped in the passenger flow image, and several objects may belong to one connected domain. Simply using the connected domains to count the passenger quantity can cause serious error. According to the analysis of passenger flow images in high-speed railway transport hub, we can classify the connected domains of passenger flow images after segmentation into two types. The first type is the large noncircular connected domains which are caused by passengers overlapped. The other type is the small connected domains which are generated by independent passenger. Based on the analysis and classification of connected domains in high-speed railway transport hub, we propose a recognition algorithm based on connected domain characters to accurately recognize the passenger flow safety in high-speed railway transport hub.

The recognition algorithm firstly classifies all the connected domains into two parts. For the independent passenger connected domains, directly calculate the passenger amount. Then, for the passengers overlapped connected domains, calculate the passenger amount by quasi-circle analyzing and matching. Finally, sum the passenger amount of two parts, and obtain the total passenger quantity in the surveillance image.

The notations of algorithm are defined as follows. n is amount of connected domains. i is the number of connected domains. S_i is the area of the i connected domain. L_i is the circumference of the i connected domain. M_i is the measurement of the i connected domain. D_s, D_l, D_m are thresholds, $D_s > D_l$. n_{i1}, n_{i2} are passenger amount of two types connected domains. n_i is the passenger amount of the i connected domain. n_q is the amount of quasi-circulars. N is the passenger amount in the image. The algorithm process is shown in Fig.3.

Main steps of the algorithm are described as follows.

Step 1: mark all the connected domains by the sequential algorithm, go to step 2.

Step 2: calculate the area of the i connected domain S_i by Eq.2

$$S_i = (B(C_i) - T(C_i)) \times (R(C_i) - L(C_i)) \quad (2)$$

Where $T(C_i)$ is the top boundary value of the i connected domain, $B(C_i)$ is the bottom boundary value of the i connected domain, $L(C_i)$ is the left boundary value of

the i connected domain, $R(C_i)$ is the right boundary value of the i connected domain, and $T(C_i) < B(C_i)$, $L(C_i) < R(C_i)$.

Step 3: compare S_i and D_s , if $S_i < D_s$, eliminate this connected domain, let $i = i + 1$, then go to step 7. Otherwise, go to step 4.

Step 4: compare S_i and D_l , if $S_i < D_l$, let $n_{i1} = 0$, calculate the circumference of the i connected domain L_i , then calculate the measurement of the i connected domain M_i by Eq.3, then go to step 5. Otherwise, go to step 6.

$$M_i = 4\pi \cdot S_i / L_i^2 \quad (3)$$

Step 5: compare M_i and D_m , if $M_i > D_m$, let $n_{i1} = 1$, $n_i = n_{i1}$ and $i = i + 1$, then go to step 7. Otherwise, eliminate this connected domain, let $i = i + 1$, then go to step 7.

Step 6: let $n_{i2} = 0$, and calculate the amount of quasi-circulars n_1 . Then, let $n_{i2} = n_1$, $n_i = n_{i2}$ and $i = i + 1$, then go to step 7.

Step 7: compare i and n , if $i > n$, calculate the N by Eq.4, and output the passenger flow safety, the algorithm finish. Otherwise, go to step 3.

$$N = \sum_{i=1}^n n_i \quad (4)$$

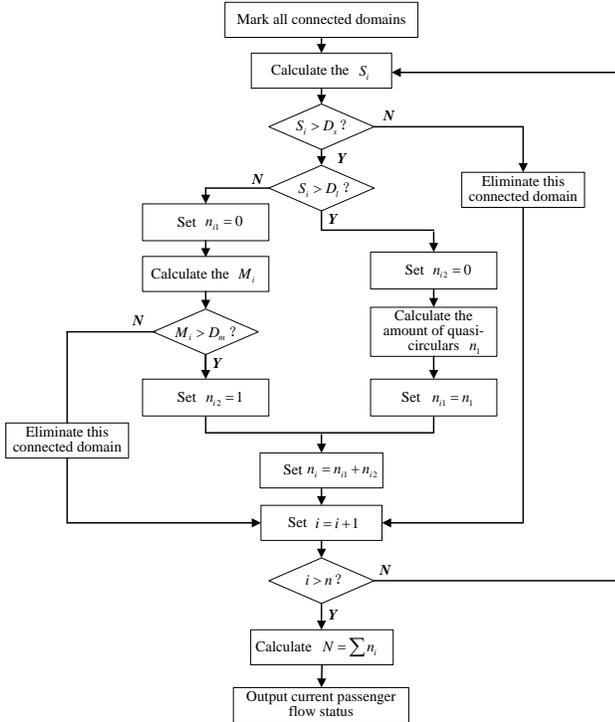


Figure 3. Recognition algorithm process

V. COMPUTATIONAL EXPERIMENTS

In this section, computational experiments are conducted to illustrate the detection algorithms proposed above. Actual sensor data from a specific Chinese high-speed railway transport hub surveillance video sequence is taken as an example. Fig. 4 is a sample of surveillance video sequence from a sensor.



Figure 4. A sample of surveillance video sequence. (a) the 1st frame image, (b) the 40th frame image, (c) the 100th frame image

A background image was built by the background model [17]. Then we used difference operators between current frame and background image to get the foreground image. The current frame, background image and foreground image are shown in Fig.5.

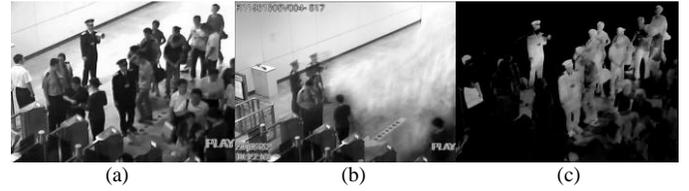


Figure 5. Current frame, background image and foreground image. (a) current frame, (b) background image, (c) foreground image

In order to reduce noise signal impact and improve the resolution of objects in foreground image, a smoothing was applied before image segmentation. We choose mean filter and median filter as smoothing methods, and make a comparison between them. The comparison is shown in Fig.6.

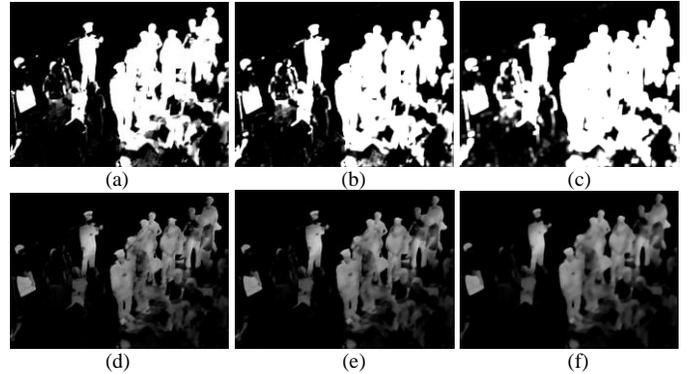


Figure 6. Comparison between mean filter and median filter. (a) 3*3 mean filter, (b) 5*5 mean filter, (c) 9*9 mean filter, (d) 3*3 median filter, (e) 5*5 median filter, (f) 9*9 median filter

As observed in Fig.6, the performance of the median filter is better than the mean filter. So we chose median filter for the foreground image smoothing. After foreground image smoothing, we used the improved watershed algorithm mentioned in section 3 for foreground image segmentation. The segmentation process is shown in Fig.7.

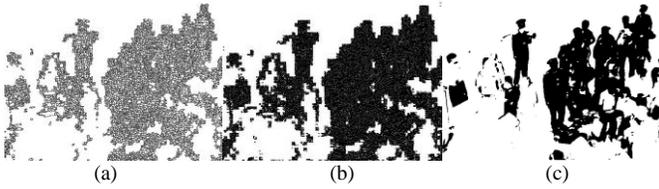


Figure 7. Segmentation process

After the foreground image segmentation, we applied the passenger flow status recognition algorithm mentioned in section 4 to recognize the passenger amount in the surveillance video. The comparison between actual passenger amount and recognition amount is shown in Fig.8, and the recognition accuracy comparison between our methods (OM) and method mentioned in Ref [18] is shown in Fig.9.

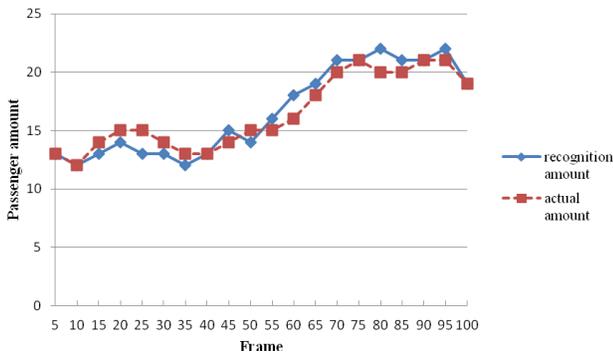


Figure 8. Comparison between actual passenger amount and recognition amount

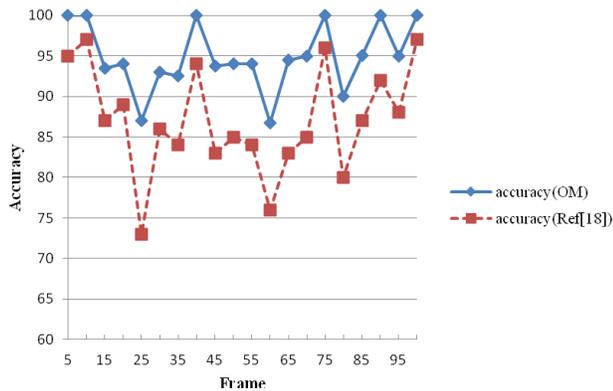


Figure 9. Comparison of recognition accuracy

As observed in Fig.8, the performance of our detection approach is satisfactory in detecting the passenger flow safety in high-speed railway transport hub. The recognition amounts in different frames are close to the actual passenger amount in surveillance video. In Fig.9, the recognition accuracy of our methods is significantly improved for the Ref [18]. After a great deal of experiments, the average accuracy of passenger flow safety detection based on our algorithm can reach 94.9%, and is better than the average accuracy of Ref [18]. The

accurate passenger flow safety detection is significant for the safety management of high-speed railway transport hub.

VI. CONCLUSION

In this paper, we considered the passenger flow safety detection in high-speed railway transport hub. According to the sensor-based detection process for passenger flow safety, an improved watershed algorithm based on tag obtained automatically was proposed for precise image segmentation, and a recognition algorithm based on connected domain characters was proposed to accurately recognize the passenger flow safety. Computational experiments on sensor data from a specific high-speed railway transport hub showed that the proposed approach was effective and efficient in detecting the passenger flow safety. The detection adaptability for different passenger flow density is a possibility for future research.

ACKNOWLEDGMENTS

This research was supported by the Fundamental Research Funds for the Central Universities (Grant no. 2015JBM044), the Talented Faculty Funds of Beijing Jiaotong University (Grant no. 2014RC005) and State Key Laboratory Program (Grant no. RCS 2016ZT016)

REFERENCES

- [1] I. Haritaoglu, D. Harwood and L Davis, "W4 : who? when? where? what? A real time system for detecting and tracking people," Proceedings of Third IEEE International Conference on Automatic Face and Gesture Recognition, pp. 222–227, 1998.
- [2] B. Benfold and I.D Reid, "Stable Multi-Target Tracking in Real-Time Surveillance Video," Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, pp. 3457–3464, 2011.
- [3] M.J Deilamani and R.N Asli, "Moving object tracking based on mean shift algorithm and features fusion," International Symposium on Artificial Intelligence and Signal Processing, pp. 48–53, 2011.
- [4] M.J.V Leach, E.P Sparks and N.M Robertson, "Contextual anomaly detection in crowded surveillance scenes," Pattern Recognition Letters, vol. 44, pp. 71–79, 2014.
- [5] S.H Choa and H.B Kang, "Abnormal behavior detection using hybrid agents in crowded scenes," Pattern Recognition Letters, vol. 44, pp. 64–70, 2014.
- [6] X.B Zhu, J Liu, J.Q Wang, C.S Li and H.Q Lu, "Sparse representation for robust abnormality detection in crowded scenes," Pattern Recognition, vol. 47, pp. 1791–1799, 2014.
- [7] G.S.K Fung, N.H.C Yung and G.K.H Pang, "Vehicle shape approximation from motion for visual traffic surveillance," Proceedings of Intelligent Transportation Systems, pp. 608–613, 2011.
- [8] H Asaidi, A Aarab and M Bellouki, "Shadow elimination and vehicles classification approaches in traffic video surveillance context," Journal of Visual Languages & Computing, vol. 25, pp. 333–345, 2014.
- [9] B Luo and J.B Xia, "A novel intrusion detection system based on feature generation with visualization strategy," Expert Systems with Applications, vol.41, pp. 4139–4147, 2014.
- [10] J.L Castro, M Delgado, J Medina and M.D Ruiz-Lozano, "Intelligent surveillance system with integration of heterogeneous information for intrusion detection," Expert Systems with Applications, vol.38, pp. 11182–11192, 2011.
- [11] D. Y Chen and P. C Huang, "Motion-based unusual event detection in human crowds," Journal of Visual Communication and Image Representation, vol. 22, pp. 178–186, 2011.
- [12] Y Cong, J.S Yuan and J Liu, "Abnormal event detection in crowded scenes using sparse representation," Pattern Recognition, vol.46, pp. 1851–1864, 2013.

- [13] K.X Dai, G.H Li, D Tu and J Yuan, "Prospects and Current Studies on Background Subtraction Techniques for Moving Objects Detection from Surveillance Video," *Journal of Image and Graphics*, vol.11, pp. 919–927, 2006.
- [14] Z.Y Xie, B.T Dong, Y Chen and Q Li, "Comparative Study on Moving Object Detection Method in the High-speed High-speed railway transport hub Video Surveillance System," *Key Engineering Materials*, vol. 467–469, pp. 503–508, 2011.
- [15] R Girisha and S Murali, "Segmentation of motion objects from surveillance video sequences using partial correlation," *IEEE Conference on image processing*, pp. 1129–1132, 2009.
- [16] Z.Y Xie, L.M Jia, Y Qin and L Wang, "A hybrid Temporal-spatio Forecasting Approach for Passenger flow safety in Chinese High-speed High-speed railway transport hub," *Discrete Dynamics in Nature and Society*, vol. 2013, pp. 1–9, 2013.
- [17] Z.Y Xie, L.M Jia, Y Qin and L Wang, "Background Modeling of Moving Object Detection in High-speed High-speed railway transport hub Video Surveillance," *Journal of Convergence Information Technology*, vol. 8, pp. 44–52, 2013.
- [18] Z.Y Xie, L.M Jia, Y Qin and L Wang, "Passenger Flow Detection of Video Surveillance: A Case Study of High-Speed High-speed railway transport hub in China," *ELEKTRONIKA IR ELEKTROTECHNIKA*, vol. 21, pp. 48–53, 2015.

City Freight Intelligent Scheduling Model Based on Genetic Algorithm

Shao Sai

School of Traffic and Transportation
Beijing Jiaotong University
Beijing, China
shaosai1218@126.com

Huang Ailing*

School of Traffic and Transportation
Beijing Jiaotong University
Beijing, China
alhuang@bjtu.edu.cn

Liu Haijun

Beijing Xinzhongyuan New Energy
Logistics Co. Ltd
Beijing, China
13910121305@qq.com

Abstract—In this paper, many similar freight demand in urban area are firstly collaborated and mixed in terms of freight classification rules. And then, with the consideration of vehicle utilization and customer logistics cost, the paper proposes a city freight intelligent scheduling model which simultaneously solve two problems of freight load and path optimization. Each customer requires the freights to be picked up and delivered by the same vehicle within a given time window. At last, the model is solved by Genetic Algorithm. A study case is introduced to show the results. The results provide a set of distribution scheme including load plan, pickup plan and delivery plan.

Keywords- City freight intelligent scheduling; Freight load; Path optimization; Genetic algorithm

I. INTRODUCTION

With development of logistics, the object of freight scheduling with high service quality, low logistics cost and economical source is recently paid attention. Freight scheduling is similar to vehicle routing problem with backhauls (VRPB) in which each customer has simultaneous pick-up and delivery demand to be satisfied, and each vehicle must complete pick-up before starting delivery. Comparing to a pure delivery or pick-up problem which ignores the simultaneous presence of deliveries and pick-ups, VRPB is more suitable for such situations occurring frequently in school buses, express delivery, grocery freight distribution and so on. VRPB has been paid much attention and solved by many approximation algorithms because VRPB is NP-hard. [1] applied an approximation algorithm to solve VRPB. [2] proposed a exact algorithm combining integer linear programming model with Lagrangian lower bound to settle VRPB in computational test with 100 customers. In [3], the situation that all customers are just served exactly once is not restricted. Moreover, an insertion-type heuristic is present for VRPB. In addition, time windows are sometimes possibly added in VRPB to become VRP with Time Windows and Backhauls (VRPTWB). [4] solved VRPTWB based on Multi-attribute label matching algorithms. [5] respectively considered VRPTWB with and without customer precedence, and use a guided local search to continually improve initial solution. The recent decade papers increasingly paid attention to used different heuristic

algorithms such as tabu algorithm [6] and genetic algorithm [7] in addressing VRPB.

Different from the previous research, the paper focuses on city freight intelligent scheduling integrating freight load and path optimization. In order to reduce logistics cost, we firstly match and collaborate similar freight demand where the features both satisfy all freight classification rules shown as follows. And then a city freight intelligent scheduling model is proposed to perform common distribution. The model considers vehicle utilization and customer logistics cost to minimize the total cost. Freight load problem and path optimization problem are simultaneously addressed. Our goal is to shorten distribution time, reduce customer logistics cost and improve resource utilization.

II. PROBLEM DESCRIPTION

There are some features such as freight weight, freight volume, pick-up node, delivery node and delivery time for each freight demand.

A. Freight Classification Rules

1) Distribution area

If the distance between two customers is too far, their freight demand will be not matched together to avoid wasting too much transport time in transit. The paper sets the same distribution area that the distance of the pick-up nodes (or delivery nodes) of any two customers must be within 20km.

2) Distribution time

The distribution time focuses on delivery time only. Similar to the distribution area, the minimum difference for the delivery time of any two customers are required within 4 hours. For example, the delivery time of customer A and B is respectively 10:00 - 12:00 and 16:10 - 17:20. The minimum difference for the delivery time of customer A and B is 4 hours and 10 minutes. The rule is not satisfied so that the freights from customer A and B cannot be mixed.

3) Freight types and vehicle requirements

For the similar freight demand which can be mixed, the freight types and vehicle requirements also need to maintain consistency. Depending on the freight types (such

as general type, refrigeration, vegetable, ultra bulky freight and so on) and the requirements the customers propose, the freights are classified and the corresponding vehicle types are chosen. The appropriate relationship between vehicle types and freights is found out under the optimal object.

B. Scheduling and Distribution

All freight demand at current time are classified under the freight classification rules. As long as satisfying both freight classification rules, the freight demand can be mixed and applied in the city freight intelligent scheduling model to get distribution plans including load plan, pick-up plan and delivery plan. Otherwise, the freight demand can be just separately loaded. Under the condition, each vehicle visit only one customer. So we just solve the freight load model to get load plan. In the end, the used vehicles implement the results. The procedure of scheduling is shown in Figure 1.

The vehicles which are responsible to transport the similar freights demand start to depart from the depot and one by one visit nodes in the same distribution area to pick up, and then successively arrive at delivery nodes. After unloading the freights, these vehicles finally come back to the depot. Moreover, during mid-tour these vehicles are not allowed to return to the depot. A simple example of common distribution is shown in Figure 2.

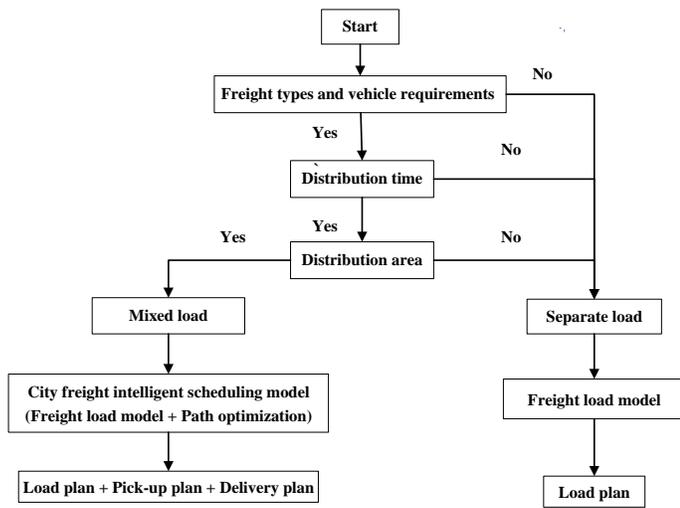


Figure 1. The procedure of scheduling.

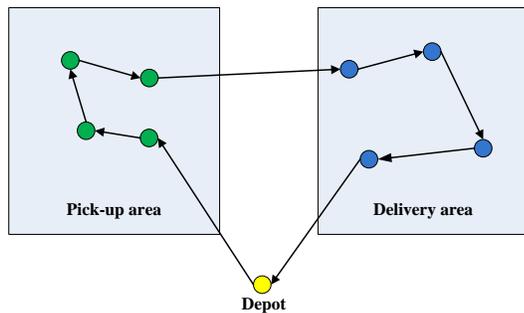


Figure 2. A simple example of common distribution.

III. CITY FREIGHT INTELLIGENT SCHEDULING MODEL

The paper develops a city freight intelligent scheduling model to simultaneously optimize load plan, pick-up plan and delivery plan for the similar freight demand. Therefore, the problem is also divided into two parts: How the freights are loaded; How and when the used vehicles visit pick-up nodes or delivery nodes. The two sub-models are introduced in detail.

As three important constraints in freight transport problem, load constraint, volume constraint and hard time window are also considered in the model. In general, the customers prefer to take over freights punctually. Therefore, hard time window just acts on delivery time in the model. In other words, there is no any requirement for pick-up time. Moreover, load constraint and volume constraint are strict during distribution.

A. Model definition

There is an directed and complete graph $G = (V, E)$. Where, node set $V = \{v_1, v_2, \dots, v_n\}$ is the combination of start depot S , pick-up node set O , delivery node set D and end depot set S' . $V = S \cup O \cup D \cup S'$. $S = \{0\}$ and $S' = \{0'\}$. The locations of the node 0 and 0' are same. The set $E = \{(v_i, v_j) | v_i, v_j \in V, i < j\}$ represents the links connecting two nodes in V . The variables in two models are defined as follows.

kk : the number of the used vehicles and the decision variable in the freight load model.

K : the used vehicle set.

w_i : freight weight of customer i (kg).

vol_i : freight volume of customer i (m^3)

l_k : the number of customers served by vehicle k .

L : the number of all customers.

W : vehicle load capacity (kg).

VOL : container load volume (m^3).

Z : total cost (yuan).

Z_1 : travel cost (yuan).

Z_2 : penalty cost (yuan).

c_t : per-unit travel cost (yuan).

c_l : per-unit late arrival penalty cost (yuan).

c_e : per-unit early arrival penalty cost (yuan).

t_k^0 : departure time of vehicle k from depot, decision variable.

$t_k^{0'}$: arrival time of vehicle k coming back to depot.

t_i^e : the earliest time of vehicle k visiting node i .
 t_i^l : the latest time of vehicle k visiting node i .
 $s_{k,i}$: pick-up node of customer i visited by vehicle k .
 $e_{k,i}$: delivery node of customer i visited by vehicle k .
 t_{ij} : travel time from node i to j .
 t_i : arrival time at node i .

R_k : the set of pick-up nodes visited by vehicle k and the decision variable in the path optimization model.

R'_k : the set of delivery nodes visited by vehicle k and the decision variable in the path optimization model.

B. Model formulation

Firstly the freight load model with the minimum number of the used vehicles is developed and solved in view of volume constraint and weight constraint. And then, the number of the used vehicles is introduced into the path optimization model.

1) Freight load model

$$\text{Minimize } Z_1 = kk \quad (1)$$

Subject to:

$$kk \leq V \quad (2)$$

$$kk = \max\left(\text{Int}\left(\frac{\sum_{i=1}^L w_i}{W}\right) + 1, \text{Int}\left(\frac{\sum_{i=1}^L vol_i}{VOL}\right) + 1\right) \quad (3)$$

Object (1) minimize the number of the used vehicles. Constraint (2) ensures the used vehicles are available. Constraint (3) calculates the number of the used vehicles takes in terms of freight weight and freight volumes. The decision variable kk represents load plan.

The freight load model is very simple and can be computed directly to get result.

2) Path optimization model

$$\text{Minimize } Z_2 = \sum_{k=1}^{kk} z_k^1 + z_k^2 \quad (4)$$

$$z_k^1 = c_i(t_k^{0'} - t_k^0) \quad (5)$$

$$z_k^2 = \sum_{i=2}^{l_k-1} \left[c_i \max\{0, t_{e_{k,i}}^l - t_{e_{k,i}}^l\} + c_e \max\{0, t_{e_{k,i}}^e - t_{e_{k,i}}^e\} \right] \quad (6)$$

Subject to:

$$R_k = \{s_{k,i} \mid s_{k,i} \in O, i=1, 2, \dots, l_k\} \quad k \in K \quad (7)$$

$$R'_k = \{e_{k,i} \mid e_{k,i} \in D, i=1, 2, \dots, l_k\} \quad k \in K \quad (8)$$

$$\sum_{k=1}^{kk} l_k = L \quad (9)$$

$$0 \leq l_k \leq L \quad k \in K \quad (10)$$

$$R_i \cap R_j = \{0\} \quad i, j \in K \quad (11)$$

$$R'_i \cap R'_j = \{0\} \quad i, j \in K \quad (12)$$

$$t_{s_{k,i+1}} = t_{s_{k,i}} + t_{s_{k,i} s_{k,i+1}} \quad k \in K, i=1, 2, \dots, l_k \quad (13)$$

$$t_{e_{k,i+1}} = t_{e_{k,i}} + t_{e_{k,i} e_{k,i+1}} \quad k \in K, i=1, 2, \dots, l_k \quad (14)$$

$$t_{s_{k,1}} = t_k^0 + t_{0 s_{k,1}} \quad k \in K \quad (15)$$

$$t_{e_{k,1}} = t_{s_{k,j_k}} + t_{s_{k,j_k} e_{k,1}} \quad k \in K \quad (16)$$

$$t_k^{0'} = t_{e_{k,j_k}} + t_{e_{k,j_k} 0'} \quad k \in K \quad (17)$$

In the path optimization model, Object (4) minimizes the total cost consisted of travel cost and penalty cost. The travel cost shown in constraint (5) is proportional to the travel time. Since the vehicles early or later arrive at delivery nodes, constraint (6) gives the corresponding penalty cost. Constraint (7) and (8) express the sets of the vehicle respectively visiting pick-up nodes and delivery nodes. Where, the pick-up node and delivery node for the same customer is required to be visited by the same vehicle. The decision variables R_k and R'_k represents pick-up plan and delivery plan, respectively. Constraint (9) and (10) ensure all customers can be served and restrict the number of the customers served by each vehicles. Constraint (11) and (12) require each pick node (or delivery node) is just visited by one vehicle once. Constraint (13)-(17) both express the arrival time (the sum of the arrival time at last node and the travel time between two node). Constraint (13) and (14) gives the arrival time at the pick-up node (or delivery node). The arrival time of the vehicle visiting first node respective in pick-up area and delivery area is present in Constraint (15) and (16). To the end, in constraint (17) the time of the vehicle coming back to depot is given.

The path optimization model belongs to NP-hard problem that computation time increases exponentially with the number of nodes in network. There are many methods to solve the problem. Exact algorithms are good choices for some comparatively simple networks and heuristic algorithms always achieve better performance for some complex networks. In order to effectively get effective solution, the paper applies Genetic Algorithm (GA) as the solution technology.

IV. STUDY CASE

A. Freight data

Assume there are totally four customers to make requests for distribution at current time. The freight demand data is shown in Table 1 and Table 2. According to the freight classification rules introduced in Section 2.1, the freights from customer 1 and 2 can be mixed and performed common distribution using the city freight intelligent scheduling model. Because the freights from customer 3 and 4 are required to be separately served, only load plans are not given in the paper.

TABLE I. FREIGHT DEMAND DATA

Customer No.	1	2	3	4
Freight type	Common	Common	Frozen	Common
Freight weight (kg)	5000	10000	8000	10000
Freight volume (m ³)	2	2	5	1
Pick-up node	X1	X2	X3	X4
Delivery node	Y1	Y2	Y3	Y4
Time window	15:00-17:00	15:30-16:00	12:00-14:00	18:00-22:00

TABLE II. THE TRAVEL TIME AND DISTANCE BETWEEN TWO NODES (MIN/KM)

Node	X1	X2	Y1	Y2
X1		8/4	72/36	80/40
X2	8/4		64/32	68/34
Y1	72/36	64/32		8/
Y2	80/40	68/34	8/4	

B. Constant parameter determination

Several constant parameters in the model are determined in terms of operation experience. $c_t = 1$, $c_l = 5$ and $c_e = 5$. The number of the available common vehicles is ten. The load capacity is 20000kg and load volume is 4m³.

C. Results and analysis

Firstly, load plan is obtained based on the freight load model. The result indicates 8 common vehicles are used. And then the path optimization model is solved by GA to obtain the pick-up plan and delivery plan. The routes, departure time at depot and arrival time at pick-up nodes or delivery nodes for the used vehicles are shown in Figure 3 and Table 3. The used vehicles depart from depot at 13:30 and successively visit node X1 and X2 to pick up for customer 1 and 2. And then, the used vehicles visit node Y1 and Y2 in delivery area, and finally come back to the depot.

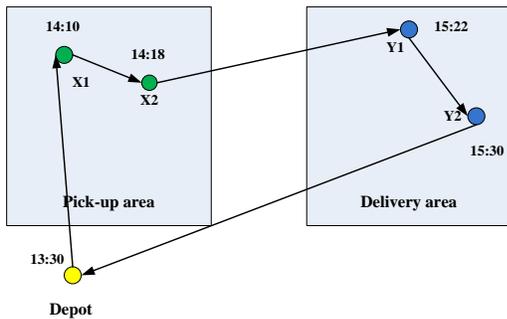


Figure 3. Pick-up plan and delivery plan

TABLE III. PICK-UP PLAN AND DELIVERY PLAN

Node order	Depot	Pick-up area		Delivery are	
		X1	X2	Y1	Y2
Time	13:30	14:10	14:18	15:22	15:30

I. CONCLUSION

The paper comprehensively considers vehicle utilization and customer logistics cost to develop the city freight intelligent scheduling model combining intelligent goods consolidation with capacity scheduling optimization for many similar mixed freight demand. The common distribution is implemented to pick up and deliver these freights in sequence. two problems, freight load problem and path optimization problem are addressed in the city freight intelligent scheduling model. In conclusion, the paper provides a set of operation scheme consisted of load plan, pick-up plan and delivery plan to the logistics company. The load plan can be obtained straightly by the freight load model. Because of being formulated as a mixed integer linear programming, the path optimization model is solved by GA to get pick-up plan and delivery plan.

ACKNOWLEDGMENT

This research was supported by National Science and Technology Support Program of China (Grant No. 2014BAH23F01), Beijing Municipal Institute of Science and Technology (Grant No. PXM2015_178215_000008) and enterprise fund (Grant No. T15L00210).

REFERENCES

- [1] Goetschalckx, M. and Jacobs-Blecha, C., "The vehicle routing problem with backhauls," European Journal of Operational Research, vol. 42(1), pp.39-51, 1989.
- [2] Toth, P. and Vigo, D., "An exact algorithm for the vehicle routing problem with backhauls," Transportation science, vol. 31(4), pp.372-385, 1997.
- [3] Wade, A.C. and Salhi, S., "An investigation into a new class of vehicle routing problem with backhauls," Omega, vol. 30(6), pp.479-487, 2002.
- [4] Cheung, R.K. and Hang, D.D., "Multi-attribute label matching algorithms for vehicle routing problems with time windows and backhauls," IIE transactions, vol. 35(3), pp.191-205, 2003.
- [5] Zhong, Y. and Cole, M.H., 2005. A vehicle routing problem with backhauls and time windows: a guided local search solution. Transportation Research Part E: Logistics and Transportation Review, 41(2), pp.131-144.
- [6] Brandao, J., 2006. A new tabu search algorithm for the vehicle routing problem with backhauls. European Journal of Operational Research, 173(2), pp.540-555.
- [7] Tasan, A.S. and Gen, M., 2012. A genetic algorithm based approach to vehicle routing problem with simultaneous pick-up and deliveries. Computers & Industrial Engineering, 62(3), pp.755-761.

Clustering and Artificial Neural Network Ensembles Based Effort Estimation

Hamdy Ibrahim^{1,2}

¹Department of Electrical and Computer Engineering,
University of Calgary, Calgary, AB, Canada

²Department of Information Systems
Menoufia University, Shebin Elkom, Egypt
hibrahi@ucalgary.ca

Behrouz H. Far

Department of Electrical and Computer Engineering
University of Calgary
Calgary, AB, Canada
far@ucalgary.ca

Abstract—Accurate effort estimation of software development projects plays a key role in project success. However, it is still a challenge activity to researchers and practitioners because of the nature of software products and dynamics in software industry and development environment. Artificial neural network (ANN) is as an effective method and has been widely used in various areas of software engineering. This paper proposes a new effort estimation method based on clustering and ANN ensembles. The contribution of the paper is twofold. First, the impact of clustering projects on the estimation accuracy is investigated. Second, the impact of using ANN ensembles instead of a single ANN is also investigated. The proposed method includes three phases called pre-processing, k-means clustering, and ANN ensembles effort estimation. The method starts with exploring the historical projects dataset. Afterward, k-means is used to cluster the projects. Finally, the proposed method as well as two other estimation methods (i.e. a single ANN and expert-based) were applied to the created clusters and results were compared using MMRE and PRED measures. The simulation results show that the proposed method significantly outperforms the two other estimation methods.

Keywords—component; Artificial Neural Network Ensembles; Clustering; K-means; Effort Estimation; Mean Magnitude Relative Error; Percentage of Predictions.

I. INTRODUCTION

The estimation of software development resources is the process of predicting the amount of effort, schedule, and cost needed to develop or maintain a software system [1]. It plays an essential role in software project management. It can be used to (1) set the budget and schedule, (2) support build versus buy decision, (3) as a part of tradeoff analysis among cost, schedule, and scope triangle, (4) provide the cost part of cost-benefit analysis, and (5) support the risk analysis of software cost and schedule [2]. Inaccurate estimation leads to software project crisis in terms of delivery delays and cost over-runs although releasing software systems on time and within the budget is a critical need for organizations.

An analysis study of around 50,000 projects (i.e. 60% from USA, 25% from Europe, and 15% from the rest of the world) developed between 2003 and 2012 reports that around 39% (i.e. 19,500) of projects were delivered on time, within the planned budget, and meeting stakeholders' needs; 43% (i.e.

21,500) of them were late, over budget, and/or partially meeting stakeholders' needs; and 18% (i.e. 9,000) of projects totally failed (cancelled prior to completion or delivered and never used) [3].

Underestimating software development effort can lead to understaffing, schedule and cost overruns, and low quality of software systems therefore can have severe impact on business reputation, credibility with customers, competitiveness, and performance. Furthermore, overestimating software development effort can lead to poor allocation and ineffective use of resources (i.e. wasted resources) [4, 5]. Resource estimation in software development project is a challenge task and more difficult than resource estimation in other industries because (1) software requirements and development tasks are ambiguous and not granularly specified, (2) there is dependency between software requirements, (3) fast evolution of technologies, and (4) the information used in the estimation process is uncertain. Addressing these challenges requires the use of a systematic estimation process.

The major role of software resource estimation in the project success and the associated challenges motivates researchers to (1) study software resource estimation problem, (2) propose new estimation techniques or combine existing techniques to better estimate effort, schedule, and cost, and (3) conducting comparative studies of current software resource estimation techniques to determine the most accurate technique. There are a vast number of estimation techniques which was proposed in the literature and can be classified into two main categories [6]:

- 1) Expert based estimation techniques where experts with relevant experiences in the application domain and in similar projects are consulted to estimate effort, schedule, and cost. This category has the following disadvantages: (1) It lacks the use of analytical methods, (2) Estimates are prone to errors and very subjective particularly when there is a mismatch between expert experience and project characteristics, (3) Sometimes, it is hard if not impossible to find experts with the right experience and If they are found, the cost will be high.
- 2) Model-based estimation techniques which use some algorithms and models to analyze historical data of past

projects to estimate effort, schedule, and cost of the new projects. This category includes statistical regression based models, machine learning based models (e.g. artificial neural networks (ANN), decision trees, case-based reasoning, Bayesian networks, etc.). It has a number of advantages over expert-based techniques. For example, the estimates are more objective, the same technique with/without adaptation can be used in different projects, and historical data of enormous number of past projects can be used to improve the accuracy of the estimation technique.

For decades, many techniques have been used for estimating software project effort, project duration, and product size. These estimation techniques have been developed using informal models (e.g. expert-judgement based estimation techniques), or formal models such as statistical and machine learning techniques [7-11]. Many researchers used artificial neural networks (ANNs) to estimate the software project effort because of its learning capability and its ability to model complex problems which are characterized by the existence of nonlinear relationship between problem variables. Examples of ANN-based estimation techniques are [12-19]. Feed-forward multilayer perceptron neural networks with back-propagation have been widely used to estimate project effort [12-14]. A number of estimation techniques were developed using radial basis function (RBF) neural networks [15]. Few research attempts [20] investigated the use of clustering and ANN ensembles and their impact on improving the estimation accuracy.

There are numerous software estimation techniques but the accurate effort estimation is still a challenge for the researchers and practitioners particularly project managers. In this paper, clustering and ANN ensembles based estimation method is proposed. The proposed method has three main phases: (1) Pre-processing, (2) K-means clustering, and (3) ANN ensembles-based estimation. The proposed method was evaluated using three datasets [21] and its performance was compared to the performance of a single ANN and expert-judgement based estimation techniques. The simulation results show that the proposed method outperforms other estimation techniques. Specifically, clustering projects improves the estimation accuracy. In addition, Applying ANN ensembles to the clustered projects also significantly improve the estimation accuracy.

The rest of the paper is organized as follows. Section 2 presents the proposed estimation method. Section 3 discusses the two measures used to evaluate the proposed method. In section 4, the proposed method is evaluated as well as the simulation results are presented and discussed. Section 5 presents the conclusion and summarizes the future research directions.

II. THE PROPOSED EFFORT ESTIMATION METHOD

The main idea of the proposed technique is using

1. Clustering to categorize the projects according to their similarity. The created clusters will be used to train and validate the neural network component. Clustering is used to improve the reliability of effort estimation because only

relevant and homogenous projects are used to estimate project effort.

2. Artificial Neural network (ANN) ensembles where multiple neural network models are created, trained, and validated to estimate the project effort. The estimated efforts are then combined and averaged to represent the predicted project effort. Neural network ensembles are used to improve the accuracy of effort estimation.

The conceptual diagram of the proposed method is shown in Figure 1. The following subsections discuss pre-processing, K-means, and ANN-based Effort Estimation phases.

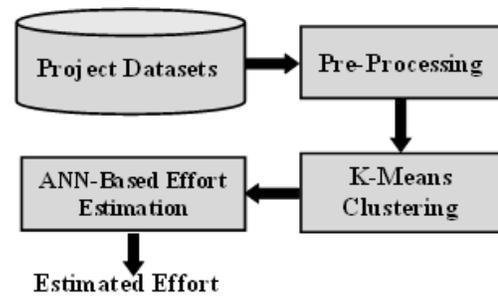


Figure 1. Conceptual Diagram of the Proposed Estimation Technique

A. Pre-Processing

In this phase, project datasets are explored to determine (1) number of projects in each dataset, (2) the key features (e.g., application domain) of the projects, (3) effort multipliers that are common between the projects, and (4) data types of the effort multipliers (e.g. numeric, discrete, etc.). In this phase, datasets are also searched for incomplete projects which have missing values for effort multipliers. According to the relative impact of the effort multipliers with missing values, a decision of including or excluding these projects is made. Conversion of discrete values to numeric values is also done to facilitate the combination of the project datasets for the purpose of validating the proposed method (see section IV-A).

B. K-means Clustering

K-means clustering technique is selected to cluster the projects because it is relatively scalable and efficient in processing large datasets. The projects are clustered as follows.

Input: A set of K-software development projects with r-features, C: number of clusters to be created where $C \leq K$. $C=K$ means each cluster will have only one project this is unrealistic.

Processing: K-means will organize the projects into C-clusters so that the similarity is high within the clusters and low between clusters. Assuming $C=3$ (i.e., 3 clusters will be formed), 3 projects each of which initially represents a cluster mean or center, will be randomly selected. Each of the remaining projects (K-3) is assigned to a cluster based on the distance between the project and the cluster mean. The project is basically assigned to the nearest (i.e., the lowest distance) cluster. Then, the mean value (i.e., center) for each cluster is recalculated according to the current projects in the cluster. The projects will be reassigned to the clusters according to the

updated cluster centers and based on which cluster center is the closest. The process will be repeated till no reassignment of projects in any cluster occurs.

Output: C clusters are created.

C. ANN Ensembles-based Effort Estimation

The ANN model used in the proposed method is shown in Figure 2.

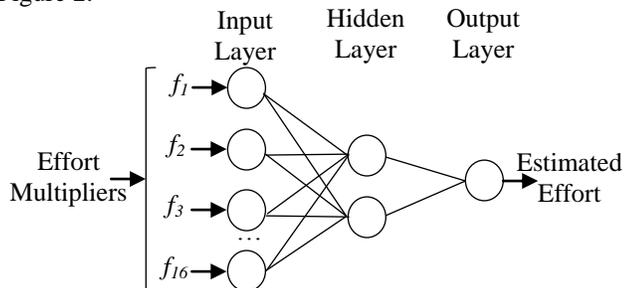


Figure 2. ANN Model

The input layer has 16 nodes each of which corresponds to an effort multiplier. The hidden layer has 2 nodes and the output layer has one node.

The ANN-based effort estimation algorithm shown in Figure 3 was developed around neural network ensembles concept. This is because the performance of neural networks and the accuracy of effort estimation can be impacted by how the data is split into training and validation datasets. Different splits of training and validation datasets produce different effort estimations. Therefore, 20 ANN models (i.e. ANN ensembles) with different training (80%) and validation (20%) splits were used to estimate the project effort. During the neural network training, weights are continuously adjusted till reaching their optimal values where the mean square error (MSE) is less than or equal to a threshold value (T_v). Datasets may contain numerous local minima. A widely used technique to overcome the problem of local minima is training ANN more than once starting with different random weights. The best ANN which has with the lowest MRE is selected because it represents most likely the global optimal values of ANN weights. In the proposed method, Each ANN ensemble is trained several times (e.g., up to 25 runs) using different random weights for the same dataset. Training is repeated until achieving an acceptable accuracy (e.g., MRE is less than or equal 10%) or reaching the maximum number of runs (e.g., 25 runs). If 25 runs are completed and MRE is still more than 10%, the project effort estimated by the ANN having the lowest MRE will be recorded. The 20 estimated efforts were combined using a weighted average technique as follows:

1. A weight which ranges from 0 to 1 is assigned to each ANN ensemble depending on its magnitude relative error (MRE) which measures the difference between actual and estimated effort for a given project relative to the actual effort. The lower MRE is the higher weight is assigned to the ANN ensemble. The MRE and weight of a ANN ensemble is calculated as follows:

$$MRE_{i,j} = \frac{|A_j - E_{i,j}|}{A_j} \quad (1)$$

$$w_i = 1 - MRE_{i,j} \quad (2)$$

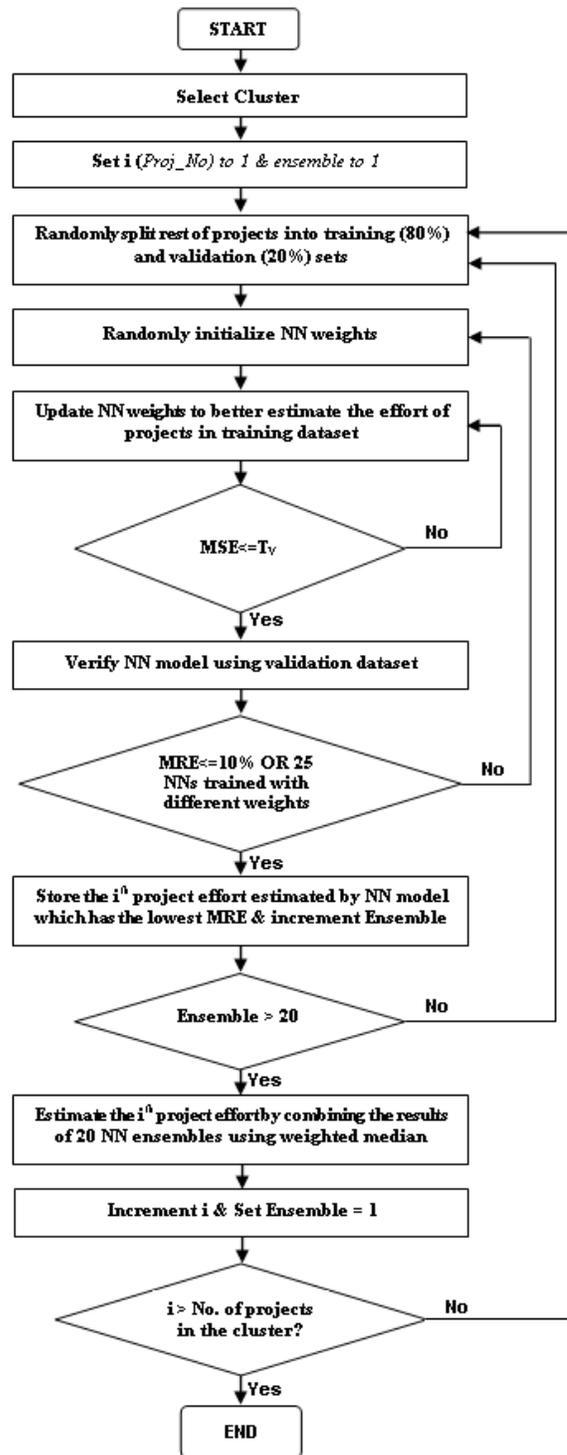


Figure 3. ANN Ensembles Estimation Process

Where:

$MRE_{i,j}$: Magnitude relative error of ANN ensemble i for project j .

A_j : Actual effort for project j .

$E_{i,j}$: Effort for project j estimated by ensemble i .

w_i : weight assigned to ANN ensemble i .

For example, a weight of 1 is assigned to the ANN ensemble whose MRE is 0 (i.e., the estimated project effort equals to the actual project effort).

2. The weighted median effort of a project j (\overline{E}_j) is calculated as follows:

$$\overline{E}_j = \frac{1}{N} \sum_{i=1}^N w_i E_{i,j} \quad (3)$$

Where:

N: Number of ANN ensembles.

w_i : weight assigned to ANN ensemble i .

$E_{i,j}$: Estimated effort for project j by ANN ensemble i .

III. PERFORMANCE EVALUATION MEASURES

Two measures are used to evaluate the performance of the proposed estimation technique.

1. Mean Magnitude Relative Error (MMRE): It is the most widely used criterion for evaluating the performance of software estimation models. MMRE is calculated as follows:

$$MMRE_k = \frac{1}{k} \sum_{j=1}^k MRE_j \quad (4)$$

Where:

K: number of projects for which the effort is estimated

MRE_j: Magnitude relative error for project j

Suppose that there are 2 estimation models with MMRE₁ and MMRE₂ respectively. If MMRE₂ is less than MMRE₁ then the performance of the second estimation model is higher than the performance of the first estimation model. Therefore, the lower the MMRE is the higher the performance of the estimation model is.

2. Percentage of Prediction (PRED): PRED(x) refers to the percentage of the projects for which MRE is less than or equal to x . PRED(x) is calculated as follows:

$$PRED(x) = \frac{S}{K} \quad (5)$$

Where:

S: Number of projects with MRE less than or equal x

K: Total number of projects

The ideal value of x which is used in software estimation technique is 0.25. Therefore, PRED(0.25) is used to compare between the proposed estimation method and other estimation methods (i.e. single ANN and expert-based estimation).

IV. EVALUATION AND RESULTS DISCUSSION

In this section, the performance of the proposed estimation technique is evaluated and compared with (1) ANN-based technique and (2) Expert-based technique. The section is divided into three subsections as follows:

- 1- *Datasets* where the characteristics of the three datasets used in the method evaluation are summarized. Furthermore, the process of combining the three datasets into a large dataset is discussed.
- 2- *Clustering* where the combined datasets are clustered into 3 clusters using K-means clustering technique. Results of clustering are presented.
- 3- *Effort Estimation* where the effort is estimated using ANN ensembles, ANN, and expert methods for the three clusters and the combined datasets. MMRE and PRED(0.25) measures are used to compare between the three estimation methods.

A. Datasets [21]

Three public datasets from PROMISE Software Engineering Repository [21] are used to evaluate the performance of the proposed method. Characteristics of the datasets are summarized in Table 1.

TABLE I. CHARACTERISTICS OF THE 3 DATASETS

Dataset		No. of Projects		No. of Effort Multipliers		
COCOMO81		63		16		
COCOMO-NASA		60		16		
COCOMO-NASA2		93		16 + 7 General Attributes		
Effort Multiplier	Description	MIN	MAX	MEAN	STD DEV	Variance
Rely	Required software reliability	0.75	1.4	1.12	0.16	0.02
Data	data base size	0.94	1.16	1.01	0.08	0.01
CPLX	process complexity	0.7	1.65	1.16	0.16	0.02
Time	time constraint for CPU	1	1.66	1.14	0.17	0.03
STOR	main memory constraint	1	1.56	1.13	0.16	0.03
VIRT	machine volatility	0.87	1.3	0.95	0.11	0.01
Turn	turnaround time	0.87	1.15	0.96	0.09	0.01
ACAP	analysts capability	0.71	1.46	1.08	0.18	0.03
AEXP	Application Experience	0.82	1.29	1.06	0.12	0.02
PCAP	Programmers Capability	0.7	1.42	1.06	0.16	0.03
VEXP	virtual machine experience	0.75	1.21	1	0.08	0.01
LEXP	language experience	0.7	1.14	1.03	0.07	0.004
MODP	modern programing practices	0.82	1.24	1.03	0.11	0.01
TOOL	use of software tools	0.83	1.24	1.02	0.1	0.01
SCHED	schedule constraint	1	1.23	1.03	0.05	0.002
LOC	Line of code	0.9	1150	83.72	135.69	18413.03

The three datasets have 16 common effort multipliers. In COCOMO81 and COCOMO-NASA datasets, the 16 effort multipliers are numeric. However, in COCOMO-NASA2, the effort multipliers have discrete values ranging from very low to extra high. In pre-processing step, the numeric-to-discrete conversion table provided by [21] is used to convert discrete values to numeric. The numeric-to-discrete conversion is required to combine the three datasets. A total of 216 projects form the combined dataset. There are two types of correlation between the effort multipliers and the effort.

- *Positive correlation*: when the effort multiplier increases, the effort also increases and vice versa. For example by increasing the software reliability (Rely) or the

complexity of the process (CPLX), the effort will increase.

- *Negative correlation*: when the effort multiplier increases, the effort decreases and vice versa. Increasing analyst’s capability (ACAP), programmer’s capability (PCAP), language experience (LEXP), or application experience (AEXP) decrease the effort.

B. Clustering

NCSS data analysis tool [22] is used to cluster the combined datasets into 3 clusters using K-means technique. Figure 4 shows a screenshot for NCSS tool. The summary of clusters created using K-means is shown in Table 2 and Table 3 shows the means of effort multipliers in the three clusters.

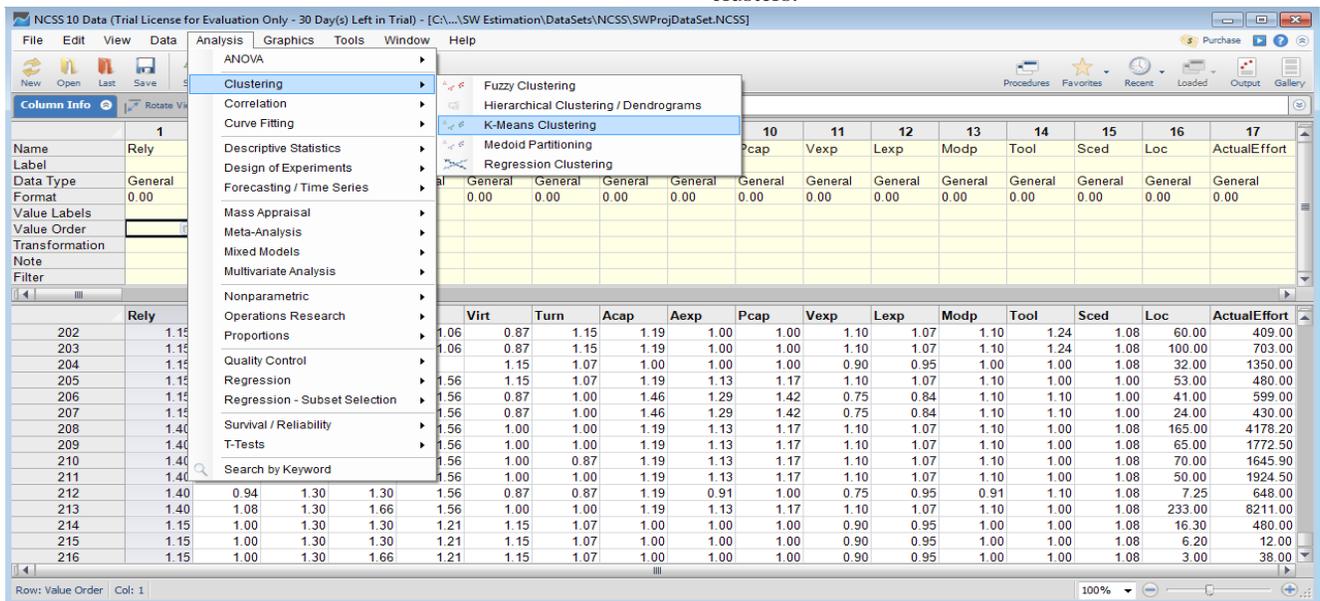


Figure 4. NCSS Data Analysis Tool

TABLE II. CLUSTERS SUMMARY

Cluster Number	Number of Projects	Percentage (%)
1	100	46%
2	80	37%
3	36	17%
Combined Dataset	216	100%

TABLE III. MEANS OF EFFORT MULTIPLIERS IN THE 3 CLUSTERS

Effort Multiplier	Cluster 1	Cluster 2	Cluster 3
Rely	1.14	1.05	1.19
Data	0.98	1.01	1.10
CPLX	1.19	1.11	1.23
Time	1.11	1.07	1.36
STOR	1.11	1.07	1.28
VIRT	0.97	0.94	0.93
Turn	0.94	0.95	1.05
ACAP	0.94	1.15	1.28
AEXP	0.97	1.14	1.12
PCAP	0.95	1.18	1.09
VEXP	1.01	1.00	0.96
LEXP	1.03	1.02	1.04
MODP	1.02	1.00	1.13
TOOL	1.03	0.96	1.15
SCHED	1.04	1.02	1.04
LOC	39.49	135.88	90.67

Table 4 presents the distance between the first 20 projects and the center of the three clusters. Distance 1 represents the distance between a project and the center of cluster 1. For example, (1) Project 1 is assigned to cluster 2 because the distance between project 1 and center of cluster 2, is the smallest distance, (2) Project 3 is assigned to cluster 1 because the distance between project 3 and center of cluster 1 is the smallest distance, and (3) Project 18 is assigned to cluster 3 because the distance between project 18 and center of cluster 3 is the smallest distance. Smallest distances for all projects are highlighted in grey.

TABLE IV. DISTANCE TO CENTER OF THE 3 CLUSTERS

Project No.	Cluster	Distance 1	Distance 2	Distance 3
1	2	5.92	5.14	5.64
2	2	4.91	4.56	5.55
3	1	4.63	4.89	6.63
4	2	6.28	5.26	6.39
5	1	3.90	4.31	5.53
6	2	6.10	5.05	5.65
7	2	4.12	3.73	6.41
8	1	6.47	7.81	6.94

9	1	3.12	4.56	5.25
10	1	4.30	5.88	5.53
11	1	4.30	5.88	5.53
12	1	2.56	4.51	5.60
13	1	3.71	5.29	6.63
14	1	6.80	8.41	7.10
15	1	6.10	7.11	7.60
16	1	3.85	5.26	5.04
17	1	4.02	5.75	5.50
18	3	4.42	4.92	3.70
19	2	8.74	8.36	9.55
20	1	4.90	6.04	5.77

Figure 5 shows a bivariate plot example which graphs the relationship between reliability (RELY) and process complexity (CPLX) effort multipliers. The bivariate plot helps to identify the degree and pattern relation between the two effort multipliers within different clusters. In cluster 1, increasing the process complexity reduces the software reliability and vice versa.

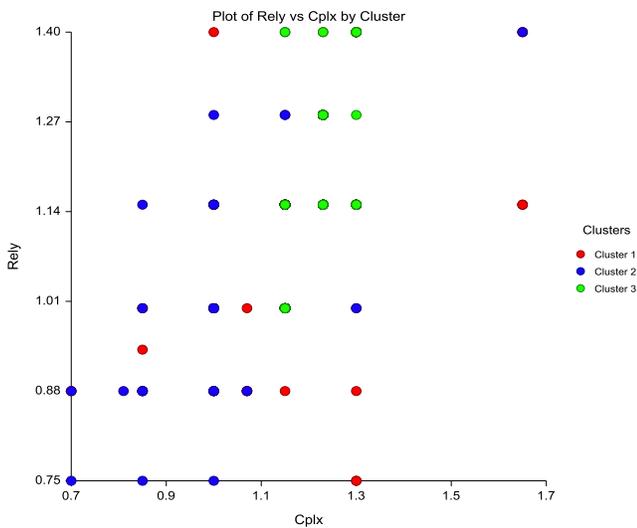


Figure 5. RELY vs. CPLX Bivariate Plot by Clusters

C. Comparison between ANN ensembles, ANN, and Expert based method

Before presenting and discussing the comparison results between the three estimation methods, we have noticed that

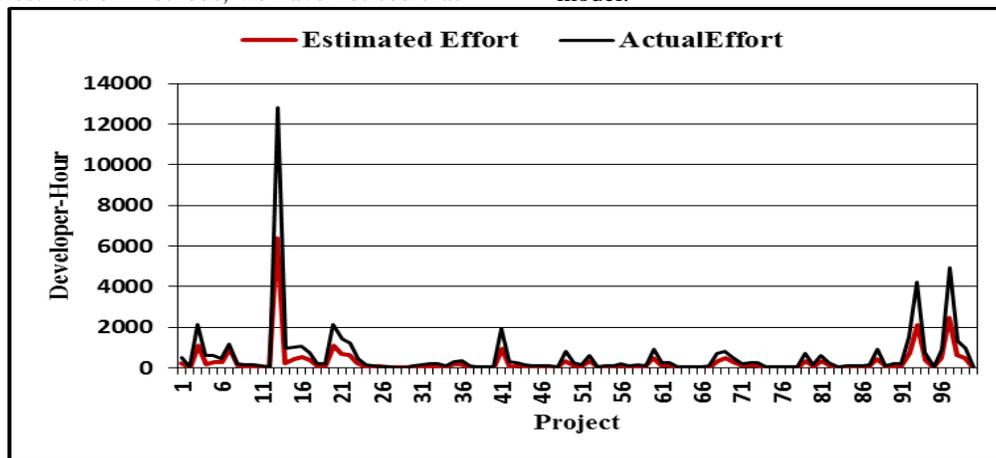


Figure 8. Estimated Effort versus Actual Effort in Cluster 1

the most influential effort multipliers vary from cluster to another (see Figures 6 and 7). For example In the combined datasets, the most influential effort multipliers are SCHED, ACAP, MODP, TOOL, LOC, VIRT, and TIME. However, the most influential effort multipliers in cluster 1 are LOC, VEXP, RELY, and AEXP. These multipliers have impact on project effort estimation more than the rest of 16 multipliers. Figure 8 shows the estimated effort versus the actual effort for 100 projects which belong to cluster 1.

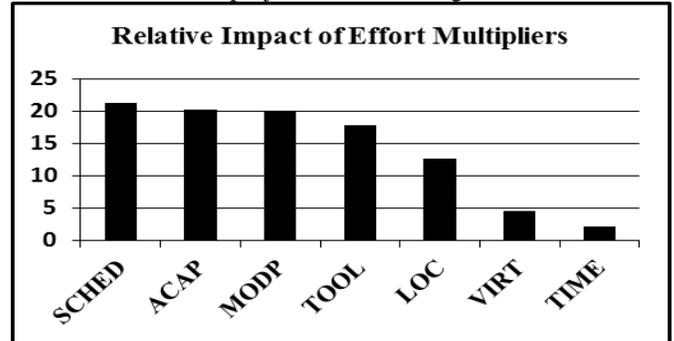


Figure 6. Relative Impact of Effort Multipliers in the Combined Datasets

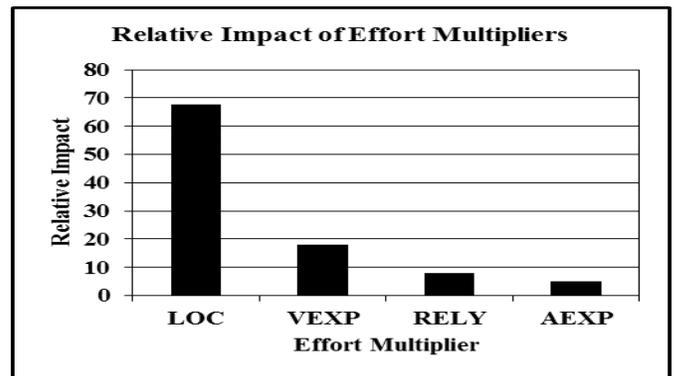


Figure 7. Relative Impact of Effort Multipliers in Cluster 1

The comparison between ANN ensembles, ANN, and expert estimation techniques using MMRE and PRED(0.25) measures is presented in Table 5.

The simulation results show that the proposed method performs better than using a single ANN and expert-based model.

TABLE V. MMRE AND PRED(0.25) SUMMARY

Dataset/ Estimation		ANN Ensembles	ANN	Expert
Combined Dataset	MMRE	0.451	0.71	1.86
	PRED(0.25)	36.31%	27.2%	19.51%
Cluster 1	MMRE	0.115	0.322	1.05
	PRED(0.25)	82%	70.65%	21.32%
Cluster 2	MMRE	0.164	0.34	0.73
	PRED(0.25)	71.45%	69.1%	29.3%
Cluster 3	MMRE	0.35	0.403	0.30
	PRED(0.25)	55.25%	51.5%	67.22%

The best results of the MMRE and PRED (0.25) measures are achieved by applying the ANN ensembles to projects in cluster 1. The values of MMRE and PRED(0.25) are 0.115 and 82% (82 projects out of 100 have MRE which is less than or equal 0.25) respectively. Overall, ANN ensembles perform better when it is applied to cluster 1. The worst results of the ANN ensembles and ANN was achieved when it is used to estimate the effort of projects belonging to the combined datasets. The estimation accuracy using ANN ensembles and ANN methods is decreased when it is trained and validated using heterogeneous projects or small number of projects. Expert-based model performs well when it is applied to a cluster with a small number of projects. Overall, clustering significantly improves the estimation accuracy and the performance of the three estimation methods. In addition, clustering the projects then using ANN ensembles to estimate the effort also enhances the estimation accuracy.

V. CONCLUSION AND FUTURE WORK

In this paper, we proposed an effort estimation method. The proposed method has been developed using clustering and artificial neural network (ANN) ensembles. It consists of three phases: pre-processing, k-means clustering, and ANN ensembles based effort estimation. The paper briefly discussed the three phases and the used estimation algorithm. NCSS data analysis tool was used to cluster the combined datasets into 3 clusters using k-means clustering method. The proposed ANN ensembles estimation algorithm, a single ANN and expert-based methods were applied to the combined datasets and the created three clusters. Two measures (i.e. MMRE and PRED(0.25)) were used to evaluate performance of the proposed estimation method and compare it to the performance of ANN and expert based estimation methods. Overall, the simulation results show that:

- 1- The proposed estimation method outperforms the ANN and expert based estimation methods.
- 2- The best results for the proposed method was achieved when it is applied to cluster 1 because cluster 1 has homogenous and/or similar projects and the number of projects is fair enough to train and validate the ANN ensembles.
- 3- The performance of the proposed method was low when it is applied to the combined datasets and cluster

3 because the combined datasets has a large number of heterogeneous projects and cluster 3 does not have enough projects to train and validate the ANN ensembles.

- 4- Clustering projects then using ANN ensembles, ANN, or expert-based estimation significantly improves the estimation accuracy.

Our future research will focus on:

- 5- Investigating the impact of using a subset of relevant effort multipliers on the accuracy and performance of the proposed estimation method.
- 6- Studying the impact of number of nodes in the hidden layer as well as number of hidden layers on the performance and accuracy of the proposed estimation method.
- 7- More experimentation using different datasets and comparing the results of the proposed estimation method with other estimation techniques proposed in the literature.

ACKNOWLEDGMENT

The authors would like to thank Electrical and Computer Engineering department at the University of Calgary, Alberta, Canada, for their logistics and financial support. The authors would like to thank Faculty of Computers and Information, Menoufia University, Egypt for their support.

REFERENCES

- [1] Magne Jørgensen, "What We Do and Don't Know about Software Development Effort Estimation," IEEE Software, Vol. 31 (2), March-April 2014, pp. 37-40.
- [2] Barry W. Boehm and Richard E. Fairley, "Software Estimation Perspectives," IEEE software, November-December 2000, pp. 22-26
- [3] J. S. Chou and C. C. Wu, "Estimating software project effort for manufacturing firms," Computers in Industry, Vol. 64 (6), August 2013, pp. 732-740.
- [4] Lionel C. Briand and Isabella Wiczorek, "Resource Estimation in Software Engineering," Encyclopedia of Software Engineering, 2002
- [5] Dirk Basten and Ali Sunyaev, "Guidelines for Software Development Effort Estimation," IEEE Computer Society, Vol. 44 (10), October 2011, pp.88-90.
- [6] Cuauhtémoc López-Martín, "Predictive Accuracy Comparison between Neural Networks and Statistical Regression for Development Effort of Software Projects," Journal of Applied Soft Computing, Vol. 27, February 2015, pp. 434-449.
- [7] Cuauhtémoc López-Martín and Alain Abran, "Neural Networks for Predicting the Duration of New Software Projects," Journal of Systems and Software Volume 101, March 2015, pp. 127-135.
- [8] T. Halkjelsvik and M. Jørgensen, "From Origami to Software Development: A Review of Studies on Judgment-Based Predictions of Performance Time," Psychol Bull, Vol. 138 (2), 2012, pp. 238-271.
- [9] C. López-Martín, C. and A. Abran, "Applying Expert Judgment to Improve an Individual's Ability to Predict Software Development Effort," International Journal of Software Eng. And Knowledge Eng. (IJSKE), Vol. 22 (4), 2012, pp. 467-483.
- [10] J. Wen, S. Li, Z. Lin, Y. Hu, and C. Huang, "Systematic Literature Review of Machine Learning Based Software Development Effort

- Estimation Models," *Information Software. Technologies*, Vol. 54 (1), 2012, pp. 41–59.
- [11] M. Jørgensen, "A Review of Studies on Expert Estimation of Software Development Effort," *Journal of Systems and Software*, Vol. 70, 2004, pp. 37-60.
- [12] J. Kaur, et al., "Neural Network-A Novel Technique for Software Effort Estimation," *International Journal of Computer Theory and Engineering*, Vol. 2, 2010, pp. 17-19.
- [13] C. S. Reddy and K. Raju, "A concise Neural Network Model for Estimating Software Effort," *International Journal of Recent Trends in Engineering*, Vol. 1, 2009, pp. 188-193.
- [14] I. Attarzadeh and S. H. Ow, "Software Development Cost and Time Forecasting Using a High Performance Artificial Neural Network Model," *Intelligent Computing and Information Science*, Vol. 134, 2011, pp. 18-26.
- [15] A. Idri, et al., "Design of Radial Basis Function Neural Networks for Software Effort Estimation," *International Journal of Computer Science*, Vol. 7, 2010, pp. 11-16.
- [16] K. V. Kumar, et al., "Software Development Cost Estimation using Wavelet Neural Networks," *Journal of Systems and Software*, Vol. 81, 2008, pp. 1853-1867.
- [17] I. K. Balich and C. L. Martin, "Applying a Feedforward Neural Network for Predicting Software Development Effort of Short-Scale Projects," *Software Engineering Research, Management and Applications (SERA)*, 2010, pp. 269-275.
- [18] V. B. Khatibi and D. N. A. Jawawi, "Software Cost Estimation Methods: A Review," *Journal of Emerging Trends in Computing and Information Sciences*, Vol. 2, 2011, pp. 21-29.
- [19] V. B. Khatibi, et al., "Neural Networks for Accurate Estimation of Software Metrics," *International Journal of Advancement in Computing Technology*, Vol. 3, 2011, pp. 54-66.
- [20] Dinesh R. Pai, Kevin S. McFall, and Girish H. Subramanian, "Software Effort Estimation using a Neural Network Ensemble," *The Journal of Computer Information Systems*, Vol. 53 (4), July 2013, pp. 49-58.
- [21] Datasets: <http://promise.site.uottawa.ca/SERepository/datasets-page.html>.
- [22] NCSS Data Analysis Tool: <http://www.ncss.com/>.

An Algorithm for Forward Reduction in Sequence-Based Software Specification

Lan Lin, Yufeng Xue
Ball State University
Department of Computer Science
Muncie, IN 47396, USA
{llin4, yxue2}@bsu.edu

Abstract

Sequence-based software specification is a rigorous method for deriving a formal system model based on informal requirements, through a systematic process called sequence enumeration. Under this process, stimulus (input) sequences are considered in a breadth-first manner, with the expected system response to each sequence given. Not every sequence needs to be further extended by the enumeration rules. The completed specification encodes a Mealy machine and forms a basis for other activities including code development and testing. This paper presents a forward reduction algorithm for sequence-based specification. The need for such an algorithm has been identified by field applications. We used the state machine as an intermediate tool to comprehend and analyze all change impacts resulted from a forward reduction, and used an axiom system for its development. We illustrate the algorithm with a symbolic example, and report a larger case study from published literature in which the algorithm is applied. The algorithm will prove useful and effective in deriving a system-level specification as well as in merging and combining partial work products towards a formal system model in field applications.

1 Introduction

Modern software development processes for safety- and mission-critical systems rely on rigorous methods for code development and testing to support dependability claims and assurance cases that provide the justified and needed confidence [14]. *Sequence-based software specification* [24, 26, 23] is such a rigorous method developed by the University of Tennessee Software Quality Research Laboratory (UTK SQRL) in the 90's to derive a formal specification and system model from the original descriptions of functional requirements. The specification, developed at an early stage in the life cycle, is important for later phases

including code development, testing, and functional formal verification. Since its inception the method has been successfully combined with a rigorous testing method (Markov chain usage-based statistical testing) and applied to a variety of industry and government projects ranging from medical devices to automotive components to scientific instrumentation, to name a few [6, 5, 13, 25, 7, 26].

Sequence-based specification systematically examines the behavior of software in all possible scenarios of use (use cases) through *sequence enumeration* [18, 24, 26, 23]. In this process a human specifier, a developer, or a domain expert considers sequences of stimuli in a breadth-first manner, where sequences of a given length are examined in lexicographic order. For each such sequence they give the last output produced by software in response to the sequence of inputs, based on the requirements. There are two situations in which a sequence need not be further extended: either it is illegal (it cannot be applied in practice) or it reaches a system state equivalent to one reached by a shorter or lexicographically earlier sequence. Sequence enumeration stops when no sequence needs to be extended and results in a table of sequences that defines a Mealy machine [18, 24]. A significant benefit of the method is that it results in a formal specification but without requiring the developer to use a formal notation. This specification can form the basis for other activities including automated model-based testing. It can also be automatically analyzed to determine whether it has certain expected, or desirable properties; this can result in requirements errors being found.

However, the specification is usually not produced in one single pass but instead in iterations. Enumerated sequences are periodically reviewed, compared with each other, added, deleted, or modified against the requirements, as one gains more and better knowledge and understanding of the system under specification. It usually results in a lot of rework when a sequence is identified to have taken the system to the same state reached by a later sequence in length-lexicographical order that has been extended. In such cases all the specification work done in specifying the

state reached by the later sequence, as well as all its successor states, will have to be erased while the same amount of work needs to be performed on the earlier sequence due to the enumeration rules. An algorithm that enables such a “forward reduction” was needed both in theory and in practical applications to maximize automation support, preserve specification effort, and eliminate the need for a lot of re-work. Field applications had also identified the need for such an algorithm as we merged or combined partial work products that focus on different system boundaries to build system models for larger and more complex applications. This paper presents a forward reduction algorithm that we have developed for this purpose. With the new theory and accompanying tool support, more requirements and specification changes can be accommodated, and the change impacts automatically computed, analyzed, and applied across the specification. The result is an enhanced specification process.

This paper is structured as follows. In Section 2 we introduce the sequence-based specification. Section 3 describes our forward reduction algorithm with a running example. In Section 4 we describe how the algorithm was applied in a published case study and report our results. Finally, Section 5 summarizes related work and Section 6 concludes the paper.

2 Sequence-based software specification

Sequence-based software specification [24, 26, 23] is a rigorous method that systematically converts ordinary functional requirements of software to a precise specification. The specification automatically translates to a formal system model for both development and testing.

To apply the method, one first identifies a *system boundary* that defines what is inside and outside the software-intensive system. This usually consists of a list of *interfaces* between the system and the *environment* of the software. From the interfaces one further collects stimuli (inputs) and responses (outputs). *Stimuli* refer to events (inputs, interrupts, invocations) in the environment that can affect system behavior. *Responses* refer to system behaviors observable in the environment. Then one explicitly *enumerates* finite stimulus sequences (representing scenarios of use), first in increasing order of length, and within the same length lexicographically. The sequence enumeration process is guided by a few rules.

As an example let us consider a simple industrial “cooker” [1]. Requirements for the cooker controller are collected in Table 1, with each sentence numbered (tagged) for easy reference. The last three requirements whose tags begin with “D” correspond to derived requirements not originally given but identified in the specification process. We further identify all stimuli and responses in Table 2. No-

Table 1. Cooker controller requirements

Tag	Requirement
0	Consider an industrial “cooker” that is subject to various factors (external heat, internal chemical reactions, etc.) that change the pressure in the cooker. A control unit attempts to keep the pressure near a specific set point by opening and closing a pressure release valve and turning a compressor on and off, according to the following rules.
1	When started, the controller does not know the status of the valve or compressor, and there is no hardware capability to poll their status.
2	Only one output signal can be sent per input signal.
3	If both the valve and the compressor appear to need changes at the same time, since only one signal can be sent, function the valve first.
4	If the input signal says the pressure is Low, then the valve should be closed and the compressor on.
5	If the input signal says the pressure is Good, then the valve should be closed and the compressor off.
6	If the input signal says the pressure is High, then the valve should be open and the compressor off.
7	When turned off, the controller does not have time to generate any output signal.
D1	Sequences with stimuli prior to system initialization are illegal by system definition.
D2	When started, the controller does not produce any externally observable response across the system boundary.
D3	Once started, the system cannot be started again without being turned off first.

tice the last two responses introduced by the theory: the *null* response (denoted 0) representing no observable behavior across the system boundary (the software may only have an internal state update), and the *illegal* response (denoted ω) representing a sequence of inputs not practically realizable (an instance of this is the power-on event being placed after other inputs in the sequence).

Table 3 shows a sequence enumeration of the cooker controller up to Length 3. We start with the empty sequence λ , and proceed from Length n to Length $n + 1$ sequences (n is a non-negative integer). Within the same length we enumerate sequences in lexicographical order. For each enumerated sequence the human specifier makes two decisions based on the requirements:

1. They identify a unique response for the most current stimulus given the complete stimulus history. For instance, the sequence SG corresponds to: software started, followed by a pressure good reading. By Requirements 1, 2, 3, and 5, the valve should be func-

Table 2. Cooker controller stimuli and responses

Stimulus / Response	Short Name	Description	Requirement Trace
Stimulus	S	Started	1
Stimulus	L	Pressure low	4
Stimulus	G	Pressure good	5
Stimulus	H	Pressure high	6
Stimulus	O	Turned off	7
Response	ov	Open valve	0
Response	cv	Close valve	0
Response	co	Compressor on	0
Response	cf	Compressor off	0
Response	0	Null	Method
Response	ω	Illegal	Method

tioned first and closed, hence the response “cv”. A sequence is *illegal* if its mapped response is ω ; otherwise, it is *legal*.

2. They consider if the sequence takes the system to the same situation that has been encountered and explored by a previously enumerated sequence. If so they note the previous sequence in the “Equivalence” column, and treat the later sequence as *reduced* to the earlier sequence. For instance, the sequence SO corresponds to: software started and then turned off. This takes the system to the same situation as the empty sequence as in both cases software is not started yet, hence SO is reduced to λ . From this point on SO and λ will behave the same for any future non-empty sequence of inputs. A sequence is *unreduced* if no prior sequence is declared for its “Equivalence” column; otherwise, it is *reduced*. In theory we also treat an unreduced sequence as reduced to itself.

The enumeration process is constrained by the following rules:

1. If a sequence (say u) is illegal, there is no need to extend u by any stimulus, as all of the extensions must be illegal (i.e., physically unrealizable). For instance, no extensions of the sequence G are enumerated because G is an illegal sequence.
2. If a sequence (say u) is reduced to a prior sequence (say v), there is no need to extend u , as the behaviors of the extensions are defined by the same extensions of v . For instance, no extensions of the sequence SO are enumerated because SO is reduced to λ .
3. When reducing a sequence (say u) to a sequence (say v), we require v be both prior (in length-

Table 3. Cooker controller sequence enumeration up to Length 3

Sequence	Response	Equivalence	Trace
λ	0		Method
G	ω		D1
H	ω		D1
L	ω		D1
O	ω		D1
S	0		D2
SG	cv		1, 2, 3, 5
SH	ov		1, 2, 3, 6
SL	cv	SG	1, 2, 3, 4
SO	0	λ	7
SS	ω		D3
SGG	cf		5
SGH	ov	SH	2, 3, 6
SGL	co		4
SGO	0	λ	7
SGS	ω		D3
SHG	cv	SG	2, 3, 5
SHH	cf		6
SHL	cv	SG	2, 3, 4
SHO	0	λ	7
SHS	ω		D3

lexicographical order) and unreduced (otherwise we could follow the reduction chain of v and get to the sequence that is unreduced). For instance, SGO is reduced to λ and not SO.

Therefore, only legal and unreduced sequences of Length n get extended by every stimulus for consideration at Length $n + 1$. The process continues until all sequences of a certain length are either illegal or reduced to prior sequences. The enumeration becomes *complete*. This terminating length is discovered in enumeration, and varies from application to application. The cooker controller enumeration terminates at Length 5.

Application of the method is facilitated with two tools developed by UTK SQRL: Proto_Seq [2] and REAL [3]. To produce a specification in either tool, one only needs to give stimuli and responses short names to facilitate enumeration; no other notation or syntax is required. The tools enforce enumeration rules by the recommended workflow and maintain internal files (XML format) current with every action.

Table 4. An example complete and finite enumeration \mathcal{E}_{ex}

Sequence	Response	Equivalence
λ	0	
a	r_1	
b	0	
c	0	a
aa	0	λ
ab	r_2	
ac	r_3	λ
ba	0	ab
bb	r_3	b
bc	0	b
aba	0	ab
abb	r_1	ab
abc	r_2	ab

3 An algorithm for forward reduction

Suppose we have a complete and finite enumeration \mathcal{E} : *complete* because all the sequences that are both legal and unreduced have already been extended by every single stimulus, and *finite* because the total number of enumerated sequences in \mathcal{E} is finite. In Table 4 we show an example of such a complete and finite enumeration, called \mathcal{E}_{ex} , that is purely symbolic. We ignore for now the semantics associated with application domains and functional requirements just to focus on how the algorithm is derived and how it works on a concrete enumeration example. Notice that \mathcal{E}_{ex} has a stimulus set $S = \{a, b, c\}$ and a response set $R = \{r_1, r_2, r_3, 0, \omega\}$ (by theory the response set always contains the two special responses 0 and ω).

Suppose we have two enumerated sequences in \mathcal{E} that are both legal and unreduced: u and v , and v is prior to u in length-lexicographical order. Because it is a complete enumeration u and v must have been extended by every single stimulus. The extensions, if they are also legal and unreduced, must have been further extended. Depending on whether the prior sequence v is the empty sequence λ or not, we handle two different cases.

In Case 1 v is not the empty sequence. An example of this is u being Sequence b and v being Sequence a in \mathcal{E}_{ex} . Now we want to “forward reduce” Sequence v to Sequence u based on two understandings: (1) u and v should take the system to the same situation (state), and (2) u has been extended to find this particular state and its successor states and we hope to utilize this information we have obtained. Of course our enumeration rules do not allow us to reduce v to a sequence that comes afterwards in length-

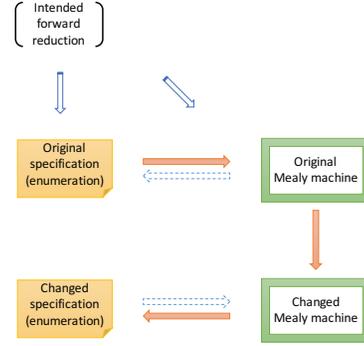


Figure 1. Our approach underlying the forward reduction algorithm

lexicographical order (namely u). What we hope to achieve is to simulate the effect of a forward reduction as if we had extended v correctly (instead of u) to identify this state and all its successor states. The question is: how could we design an algorithm that automatically computes and applies all required changes to the existing specification (enumeration) to make it happen?

As discussed in [17] a single atomic specification change (e.g., changing the response or equivalence of an enumerated sequence) could have a rippling effect on most enumeration entries and hence have a comprehensive change impact on the existing specification. The crux we found is to understand the changes utilizing state machines as an intermediate and visualizing tool, as the rigorous specifications in enumeration form have a formally defined correspondence with state machines [18], and the change impacts are more intuitively analyzed in state machines than in specifications. Figure 1 illustrates our approach.

We first give our algorithm for Case 1, and then explain it with an example.

Algorithm *CompMapForForwardRed1*(\mathcal{E}, u, v)

Input:

A complete and finite enumeration \mathcal{E} , two enumerated legal and unreduced sequences u and v such that v is prior to u and $v \neq \lambda$

Output:

A hash map κ mapping each unreduced sequence in \mathcal{E} to an unreduced sequence in \mathcal{E}' that represents the same state (if the state is preserved after the change), or nil (otherwise)

1. Initialize an empty hash map κ .
2. $\kappa(\lambda) \leftarrow \lambda$
3. $i \leftarrow 0$
4. **repeat**
5. **for** every enumerated sequence in \mathcal{E} of the form: prefix sequence p followed by stimulus s is reduced to sequence w

6. **do if** $ps \neq v$ and $w \neq u$ and $\kappa(w) = nil$ and $\kappa(p) \neq nil$ and $BlackBox(\mathcal{E}, \kappa(p)) \neq \omega$
7. **then** Let $\kappa(p)$ concatenated with s be a candidate for $\kappa(w)$.
8. **for** every unreduced sequence w that has designated candidates in steps 5-7
9. **do** $\kappa(w) \leftarrow$ its first candidate in (length-)lexicographic order
10. $i \leftarrow i + 1$
11. **if** $i = |v|$
12. **then** $\kappa(u) = v$
13. **until** The last iteration has no new sequence defined for κ .
14. **return** κ

Algorithm *ForwardReduction1*(\mathcal{E}, u, v)

Input:

A complete and finite enumeration \mathcal{E} , two enumerated legal and unreduced sequences u and v such that v is prior to u and $v \neq \lambda$

Output:

A complete and finite enumeration \mathcal{E}'

1. Initialize an empty hash map κ .
2. $\kappa \leftarrow CompMapForForwardRed1(\mathcal{E}, u, v)$
3. Initialize \mathcal{E}' to contain λ only, with λ mapped to 0 and unreduced.
4. Add sequence v in \mathcal{E}' . Map v to its response in \mathcal{E} and reduce it to v .
5. **for** every enumerated sequence in \mathcal{E} of the form: prefix sequence p followed by stimulus s mapped to response r and reduced to sequence w
6. **do if** $ps \neq v$ and $\kappa(p) \neq nil$ and $BlackBox(\mathcal{E}, \kappa(p)) \neq \omega$
7. **then** Add the following sequence in \mathcal{E}' : prefix sequence $\kappa(p)$ followed by stimulus s mapped to response r and reduced to sequence $\kappa(w)$
8. **for** every enumerated illegal and unreduced sequence w in \mathcal{E}
9. **do if** $\kappa(w) \neq nil$ and $BlackBox(\mathcal{E}, \kappa(w)) \neq \omega$
10. **then for** every stimulus s
11. **do** Add the following sequence in \mathcal{E}' : prefix sequence $\kappa(w)$ followed by stimulus s mapped to ω and reduced to sequence $\kappa(w)$
12. **return** \mathcal{E}'

A crucial step in the solution is figuring out how the state space has changed from the old automaton to the new automaton with the intended forward reduction, and what the changes have implied for the corresponding specification. As established in [18, 17] a complete and finite enumeration encodes a finite state automaton with Mealy outputs (a Mealy machine) that can be computed algorithmically. Every state of the Mealy machine corresponds to a block (equivalence class) of stimulus sequences; they all take the system to this common state starting from the initial state. Every state is represented by a unique unreduced sequence, which is the first sequence in length-lexicographical order in this block of sequences and the one you first encountered

in sequence enumeration. For instance with \mathcal{E}_{ex} the automaton has four states represented by unreduced sequences λ , a , b , and ab , respectively.

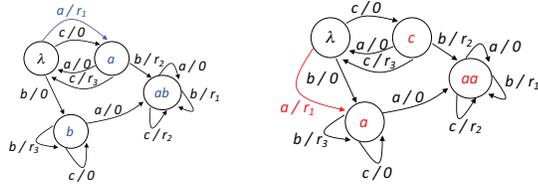
The main algorithm, *ForwardReduction1*, calls two algorithms: *CompMapForForwardRed1* and *BlackBox* (*CompMapForForwardRed1* also calls *BlackBox*), to compute the new enumeration after the change. We have “1” appended at the end of the names indicating Case 1 (“2” will be used for versions solving Case 2). Notice that *BlackBox* is an algorithm that computes the mapped response for any possible stimulus sequence, no matter whether it is explicitly enumerated in the table [18, 17], once you have a completed sequence enumeration.

CompMapForForwardRed1 computes a hash map κ that maps unreduced sequences in the old enumeration \mathcal{E} to unreduced sequences in the new enumeration \mathcal{E}' as follows: Suppose w is an arbitrary unreduced sequence in \mathcal{E} . Notice that with forward reduction no additional state will be introduced.

- If $\kappa(w)$ is nil, the state represented by w in \mathcal{E} is eliminated after the change.
- If $\kappa(w)$ is w' , the state represented by w in \mathcal{E} is preserved but represented by the unreduced sequence w' (which may or may not be the same as w) in \mathcal{E}' .

We start with the empty sequence that represents the initial state, which remains in the modified automaton with the same unreduced sequence λ (Line 2). Then we build more pairs into this mapping in iterations (Lines 4-13). With each iteration we expand the leaf nodes by one level identifying more successor states as being preserved (Lines 5-7), essentially building a spanning tree rooted in the initial state. If a successor state has more than one direct predecessors, all paths are computed and compared to select the first in length-lexicographical order as its corresponding unreduced sequence in the modified enumeration (Lines 8-9). The state previously represented by u in \mathcal{E} is represented by v in \mathcal{E}' , and the pair added at a certain iteration (Lines 3, 10-12).

Using this κ mapping we are able to construct the new enumeration \mathcal{E}' from the old one \mathcal{E} line by line. Algorithm *ForwardReduction1* first calls *CompMapForForwardRed1* to compute κ (Lines 1-2). Then it explicitly defines two special sequences in \mathcal{E}' : the empty sequence (Line 3) and the sequence v (Line 4). Next it examines each enumerated sequence in \mathcal{E} to see if the transition depicted by this row still exists in the modified automaton (for the same transition to be preserved the start state must be preserved and the transition cannot be redirected). If so a corresponding row in \mathcal{E}' is computed using κ about what new unreduced sequences now represent the two involving states (Lines 5-7). Finally some enumeration entries may need to be added manually (and not converted from old entries) due to a state



(a) Before the forward reduction (b) After the forward reduction

Figure 2. The Mealy machine before and after forward reducing a to b on \mathcal{E}_{ex}

being represented by a different unreduced sequence that has a different legality status after the change (and therefore has to be explicitly extended in \mathcal{E}') (Lines 8-11).

For our example of “forward reducing” Sequence a to Sequence b in \mathcal{E}_{ex} , applying Algorithm *CompMapForForwardRed1* gives the following mappings in κ :

Iteration 0	$\kappa(\lambda) = \lambda$
Iteration 1	$\kappa(a) = c, \kappa(b) = a$
Iteration 2	$\kappa(ab) = aa$

Figure 2 shows the automata before and after the change. We label each state with the representing unreduced sequence in the corresponding enumeration. Notice that the arc with the input/output (a/r_1) from the initial state is redirected to point to the same state reached by Sequence b (also from the initial state). Although this is the only arc redirected (all the states are preserved in the modified automaton and all other arcs remain unchanged), every state except the initial state corresponds to a newly computed unreduced sequence as a result of the change.

Except for λ and a which are defined by Steps 3-4 of Algorithm *ForwardReduction1*, all the rows in the modified enumeration \mathcal{E}'_{ex} are computed by Steps 5-7 from \mathcal{E}_{ex} on a line-by-line basis. For instance the original row for Sequence ba being mapped to 0 and reduced to Sequence ab is converted to the following row in \mathcal{E}'_{ex} : $\kappa(b)a$ being mapped to 0 and reduced to $\kappa(ab)$, i.e., aa being mapped to 0 and reduced to itself (essentially it is unreduced). Figure 3 shows how each row in \mathcal{E}'_{ex} is constructed from possibly a corresponding row in \mathcal{E}_{ex} by applying our forward reduction algorithm. For this example no new rows need to be added running Steps 8-11 of the algorithm.

Case 2 handles the situation when the prior sequence under “forward reduction” is the empty sequence. This affects how transitions are defined out of the initial state. Both algorithms are modified slightly to accommodate this. Because of the page limit we are not including detailed discussion of Case 2 here.

It is worth mentioning that our forward reduction algorithms have polynomial time complexity. They take time

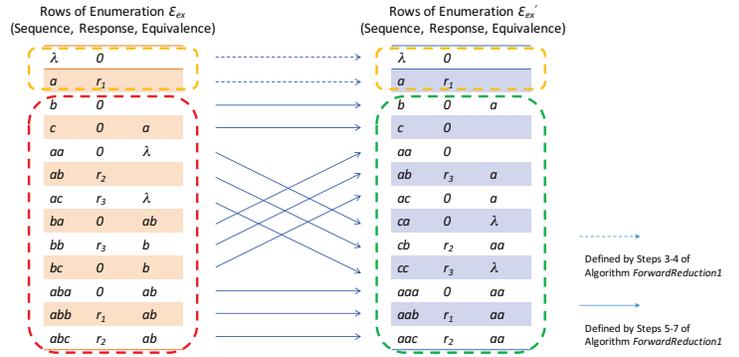


Figure 3. Constructing \mathcal{E}'_{ex} from \mathcal{E}_{ex} applying Algorithm *ForwardReduction1*

$O(n^2 \log(n))$, where n is the size of the enumeration. We hope to emphasize that our forward reduction algorithms (for both cases) were first developed in functional form using an axiom system for sequence-based specification [18] and proved for correctness before they were turned into procedural definitions that facilitate tool implementation. The algorithms were then fully implemented in a newer version of Proto.Seq that is ready for release. Using a theory-based approach ensures that our solution is backed up by sound mathematics and built on a solid theoretical foundation.

4 A case study: The mine pump controller

We did the case study of a mine pump controller software whose requirements are taken from [16]. First we developed two enumerations focusing on different system boundaries. The first enumeration includes all four human inputs from either an ordinary human operator or a special human operator (i.e., the supervisor); both could switch the pump on or off under different conditions. The second enumeration includes all six sensor inputs about the carbon monoxide and the airflow levels, the methane level, detected pump failure, and the water level. Then we merged them into one enumeration that handles all the ten system inputs using a merging procedure we defined for combining specifications. The merged enumeration contains new sequences as placeholders, which were then considered one by one by a human specifier to define the behavior involving the interaction of stimuli from the different smaller specifications. Our forward reduction algorithm was applied in defining these new entries. Table 5 shows the data we collected from the completed three specifications.

The forward reduction algorithm was applied twice in defining new sequences in the merged enumeration:

1. The new sequence **wlh.hpon** corresponds to the following sequence of events: sensor reporting water

Table 5. Data on the enumerations of the mine pump controller software

	First Enumeration	Second Enumeration	Merged Enumeration
# of Stimuli	4	6	10
Termination Enumeration Length	2	5	5
Sequences Extended	2	16	22
Sequences Analyzed	9	129	265

level between low and high limits, then an ordinary human operator switching the pump on. Based on the requirements this gets the system to the same state as a later sequence in length-lexicographical order that has been explored in one of the smaller specifications: **wlh.wah.wlh**. The later sequence corresponds to sensor reporting the water level first between low and high limits, then above the high limit, and then back to between low and high limits again. In this sequence when water level first gets above high the pump controller automatically switches the pump on (same effect as if it were turned on by an ordinary human operator) and it stays on even though water level returns to between low and high, based on the requirements. Without the forward reduction algorithm one would need to extend **wlh.hpon** and its extensions the same way they have treated **wlh.wah.wlh** and its further extensions to explore the state and all its successor states, and all the work they have done on the later sequence would have been erased. With the forward reduction algorithm all the work they have done exploring these states is preserved, and the specification is automatically modified to accommodate all change impacts.

- As a result of the above forward reduction we have in the merged enumeration **wlh.hpon.l(critical)=true** being extended. It represents the following sequence of events: water sensor reporting water level between low and high limits, an ordinary human operator switching the pump on, and then sensor reporting carbon monoxide and airflow levels critical. This gets the system to the same state as an earlier sequence introduced during the merge that has not been explored: **l(critical)=true.wlh.hpon**. The forward reduction algorithm was applied again to automatically extend this earlier sequence and make all the needed changes

to the specification, eliminating the need for any unnecessary rework.

5 Related work

Sequence-based specification emerged from the functional treatment of software as described by Mills [21, 19, 20]. The development was most directly influenced by the trace-assertion method of Parnas [22, 4] and the algebraic treatment of regular expressions by Brzozowski [8]. Foundations of the sequence-based specification method were established by Prowell and Poore in [24, 26, 23]. An axiom system for sequence-based specification was developed more recently [18] for a formal treatment. The axiom system was used in developing a theory and a set of algorithms that manage requirements changes and state machine changes [17]. The algorithm presented in this paper can be added to that set.

The primary distinction of sequence-based specification from its nearest neighbors (the *trace-assertion method* [11, 9, 10, 15, 22, 4] and *software cost reduction* [12]) is that we evolve the discovery and invent the state machine from requirements through systematic enumeration of input sequences and recording sequence equivalences based on future behavior. Likewise, the theory presented in this paper, as well as that underlying other change algorithms differs from the conventional state change theory in that it is designed for human interactive sequence enumeration and revision, and targeted at an appropriate subset of Mealy machines that can be obtained through enumeration.

6 Conclusion and future work

Sequence-based specification is a rigorous method that converts informal, imprecise descriptions of functional requirements to a precise software specification. The process is iterative in nature due to a human specifier's evolving learning and understanding of the system under specification. Field applications had identified the need for a forward reduction algorithm that could equate an earlier sequence to a later sequence in length-lexicographical order when they result in the same system state. This paper presents an algorithm we have developed for this purpose, using the automaton theory and an axiom system for sequence-based specification. We have fully implemented the algorithm in a supporting tool, which will turn out to be useful and effective not only in deriving a system-level specification but also in merging and combining partial work products towards a formal system model. We illustrate the algorithm with a symbolic example, and report a larger case study from published literature. The enhanced theory, practice, and tool support will facilitate and advance the application of sequence-based specification in field use.

Future work is along the line of scaling sequence-based software specification to very large and complex software-intensive systems. Work is under way to combine, merge, and compose smaller specifications that focus on different system boundaries. We are also conducting a number of other case studies to verify the correctness of the proposed theory and evaluate its effectiveness.

Acknowledgements

The authors would like to thank Tom Swain (previous manager of UTK SQRL) for helping set up an environment for new development and testing, and Xin Guo for earlier code contribution. This work was generously funded by Rockwell Collins, Air Force Research Laboratory, and Ontario Systems through the NSF Security and Software Engineering Research Center (S²ERC).

References

- [1] 2010. Jesse H. Poore. Private communication.
- [2] 2016. Prototype Sequence Enumeration (Proto.Seq). Software Quality Research Laboratory, The University of Tennessee. <https://sourceforge.net/projects/protoseq/>.
- [3] 2016. Requirements Elicitation and Analysis with Sequence-Based Specification (REALSBS). Software Quality Research Laboratory, The University of Tennessee. <http://realsbs.sourceforge.net>.
- [4] W. Bartussek and D. L. Parnas. Using assertions about traces to write abstract specifications for software modules. In *Proceedings of the 2nd Conference of the European Cooperation on Informatics*, pages 211–236, Venice, Italy, 1978.
- [5] T. Bauer, T. Beletski, F. Boehr, R. Eschbach, D. Landmann, and J. Poore. From requirements to statistical testing of embedded systems. In *Proceedings of the 4th International Workshop on Software Engineering for Automotive Systems*, pages 3–9, Minneapolis, MN, 2007.
- [6] L. Bouwmeester, G. H. Broadfoot, and P. J. Hopcroft. Compliance test framework. In *Proceedings of the 2nd Workshop on Model-Based Testing in Practice*, pages 97–106, Enschede, The Netherlands, 2009.
- [7] G. H. Broadfoot and P. J. Broadfoot. Academia and industry meet: Some experiences of formal methods in practice. In *Proceedings of the 10th Asia-Pacific Software Engineering Conference*, pages 49–59, Chiang Mai, Thailand, 2003.
- [8] J. Brzozowski. Derivatives of regular expressions. *Journal of the ACM*, 11(4):481–494, 1964.
- [9] J. Brzozowski. Representation of a class of nondeterministic semiautomata by canonical words. *Theoretical Computer Science*, 356:46–57, 2006.
- [10] J. Brzozowski and H. Jürgensen. Representation of semiautomata by canonical words and equivalences. *International Journal of Foundations of Computer Science*, 16(5):831–850, 2005.
- [11] J. Brzozowski and H. Jürgensen. Representation of semiautomata by canonical words and equivalences, part II: Specification of software modules. *International Journal of Foundations of Computer Science*, 18(5):1065–1087, 2007.
- [12] C. L. Heitmeyer. Software cost reduction. In J. J. Marciniak, editor, *Encyclopedia of Software Engineering*. Wiley-Interscience, 2001.
- [13] P. J. Hopcroft and G. H. Broadfoot. Combining the box structure development method and CSP for software development. *Electronic Notes in Theoretical Computer Science*, 128(6):127–144, 2005.
- [14] D. Jackson, M. Thomas, and L. I. Millett, editors. *Software for Dependable Systems: Sufficient Evidence?* National Academies Press, 2007.
- [15] R. Janicki and E. Sekerinski. Foundations of the trace assertion method of module interface specification. *IEEE Transactions on Software Engineering*, 27(7):577–598, 2001.
- [16] M. Joseph, editor. *Real-Time Systems: Specification, Verification and Analysis*. Prentice Hall International, London, United Kingdom, 1996.
- [17] L. Lin, S. J. Prowell, and J. H. Poore. The impact of requirements changes on specifications and state machines. *Software: Practice and Experience*, 39(6):573–610, 2009.
- [18] L. Lin, S. J. Prowell, and J. H. Poore. An axiom system for sequence-based specification. *Theoretical Computer Science*, 411(2):360–376, 2010.
- [19] R. C. Linger, H. D. Mills, and B. I. Witt. *Structured Programming: Theory and Practice*. Addison-Wesley, 1979.
- [20] H. D. Mills. The new math of computer programming. *Communications of the ACM*, 18(1):43–48, 1975.
- [21] H. D. Mills. Stepwise refinement and verification in box-structured systems. *IEEE Computer*, 21(6):23–36, 1988.
- [22] D. L. Parnas and Y. Wang. The trace assertion method of module interface specification. Technical Report 89-261, Queens University, 1989.
- [23] S. J. Prowell and J. H. Poore. Sequence-based software specification of deterministic systems. *Software: Practice and Experience*, 28(3):329–344, 1998.
- [24] S. J. Prowell and J. H. Poore. Foundations of sequence-based software specification. *IEEE Transactions on Software Engineering*, 29(5):417–429, 2003.
- [25] S. J. Prowell and W. T. Swain. Sequence-based specification of critical software systems. In *Proceedings of the 4th American Nuclear Society International Topical Meeting on Nuclear Plant Instrumentation, Controls and Human-Machine Interface Technology*, Columbus, OH, 2004.
- [26] S. J. Prowell, C. J. Trammell, R. C. Linger, and J. H. Poore. *Cleanroom Software Engineering: Technology and Process*. Addison-Wesley, Reading, MA, 1999.

MSECO-DEV: Application Development Process in Mobile Software Ecosystems

Awdren de L. Fontão
Institute of Computing
UFAM
Manaus – AM, Brazil
awdren@icomp.ufam.edu.br

Rodrigo Pereira dos Santos
PESC/COPPE
UFRJ
Rio de Janeiro – RJ, Brazil
rps@cos.ufrj.br

Jackson Feijó Filho
Institute of Technology
Development - INDT
Manaus – AM, Brazil
jackson.feijo@indt.org.br

Arilo Claudio Dias-Neto
Institute of Computing
UFAM
Manaus – AM, Brazil
arilo@icomp.ufam.edu.br

Abstract — In a Mobile Software Ecosystem (MSECO), the central organization (keystone), must restructure processes to aid external developers to produce mobile applications. The external developer helps the keystone to reach goals, such as growing number of mobile applications. However, there is no process in this context to support developers in the development aligned with the keystone’s goals. This paper presents MSECO-DEV, a process to support external developers in reaching keystone’s goals by developing mobile applications. MSECO-DEV comprises 8 activities, 7 artifacts, 8 recommendations, and 17 practices. Activities, recommendations, and practices were evaluated by 65 Brazilian developers (experts and novices). Such developers acted within the main MSECOs (Android, iOS and Windows Phone) to assess their benefits for the mobile applications development routine. As result, we stated that developers have difficulties to perform marketing activities, as well as to find materials that support development. Practices, activities, and recommendations were also evolved and adjusted for the definition of MSECO-DEV.

Keywords — *Software Ecosystem; Mobile; Software Process.*

I. INTRODUCTION

A MSECO comprises several elements, such as mobile applications, mobile applications store, users, keystone (i.e., central organization), and external developers [1]. Each element in the MSECO has an expected responsibility and performance that depend on the management activities performed by its keystone. The keystone needs to assure that the ecosystem keeps running and meets its goals, such as growing number of developers within the ecosystem, and of mobile applications available on the mobile application store [1].

In this context, the mobile application stores act as important MSECO instruments to make the entry of new developers and mobile applications easier. However, mobile application store and keystone cannot achieve their goals by themselves [1]. Fontão et al. [2] pointed out the following research opportunity: “*how does the quality assurance strategies depend on the specific solutions proposed to each MSECO (e.g., practices to help it in running successful, and guidelines to develop mobile apps)?*”.

As an alternative to this scenario, the MSECOs have been investing in tools, forums, and support materials. Such strategy aims to create an effective relation between developers and the

keystone [3]. In this context, there is a difficult task: balancing keystone’s and external developer’s expectations [3]. Even with the keystone’s investment in the mobile development aid, few practices and processes have been defined to support the developer’s work [1]. This situation leverages the need for customized development processes to sustain the development of high quality mobile applications that are able to remain competitive in the MSECO [4].

Based on this scenario, this paper presents a process to support mobile applications’ developers aiming to reach keystone’s goals, named MSECO-DEV. To define MSECO-DEV, we followed a methodology (detailed in Section III.A) that comprises: (1) execution of an *observational study*, (2) a *systematic mapping study*, (3) a *first evaluation* with expert developers, and (4) a *second evaluation* with novice developers. In addition to MSECO-DEV process itself, as an important contribution of this work, the methodology reported on the definition of the MSECO-DEV can serve as basis for new contributions in the software ecosystems research. This paper is organized as follows. Section II presents background. Section III describes MSECO-DEV and the methodology to build it. Section IV presents the planning, execution and results of the first survey conducted to evaluate the process with experienced developers in MSECO. Section V discusses procedures and results of the second survey conducted with novice developers to evaluate the activities and practices. Finally, Section VI concludes the paper with future work.

II. DEVELOPERS IN MSECO

The Developer eXperience (DX) consists of experiences related to all types of activities that a developer might encounter as part of their involvement in software development. We can mention as sources of DX: (a) *infrastructure development*: management and development tools, programming languages, libraries, platforms, processes, and methods; (b) *perceptions about the work*: respect and recognition; and (c) *sense of contribution*: alignment of developers’ work and contributions regarding keystone’s objectives and plans [6].

Concerning “*infrastructure development*”, the keystone needs to undertake efforts to provide developers with a framework aiming to expand the MSECO frontiers. It can be achieved with guidelines to create new apps. From the

perspective of “*perceptions about the work*”, the developer needs to create and deliver apps that reach the best existing niches of users, and gain visibility into the app store based on the quality of apps. Finally, regarding “*sense of contribution*”, the keystone has goals that can be increased by external developers as: (i) the number of ecosystem’s apps, and (ii) the star ratings of apps published in the app store. However, keystones can hardly meet the entire demand of society only with their own internal structure. Within a MSECO, external developers can help organizations in maintaining their strategy since they work directly in solving this limitation [1].

In this scenario, developers rarely use any formal development processes and have no organized tracking of their development efforts [7]. Regarding the development processes in the context of MSECO, related work that offers insights into potential alternatives to the MSECO-DEV was not found [1]. As the app development become more complex, it will be essential to adapt and/or apply software engineering processes to ensure the construction of high-quality apps [7]. For this reason, there is a need for structuring processes for managing elements of an MSECO [5]. It involves all app development activities performed by a developer, once the MSECO context is dynamic and depends on the keystone goals [6].

III. MSECO-DEV: APPLICATION DEVELOPMENT PROCESS IN MSECO

In this section, we present the MSECO-DEV (*Mobile Software Ecosystem – Development*). This process aims to improve the perception of app development by structuring a process and evaluating benefits for developers (Figure 1).

A. Methodology

The methodology to define MSECO-DEV was based on five steps. Initially, an *observational study* was conducted through training sessions within the Windows Phone MSECO with 716 developers where the keystone’s goals were driven by the growing number of downloads, apps, and developers, as detailed in [3]. This study helped us to propose MSECO Skill methodology, and to identify activities of app development in the MSECO context. With the application of MSECO Skill, the Windows Phone MSECO keystone (Microsoft) expanded the quantity and quality of mobile apps. This study concluded that there is a lack of app development approaches in the context of a MSECO to support external developers’ activities.

After running the observational study, a *systematic mapping study* of the technical literature was conducted in order to extract processes, benefits, characteristics, and areas studied in MSECO [1]. This mapping study pointed out the lack of approaches, methodologies, processes, and tools to support MSECO. We also analyzed the selected studies to extract *recommendations* (e.g., guidelines to implement activities) and *practices* (e.g., exercises to achieve concrete results regarding the keystone’s goals) that might be associated with the MSECO-DEV activities.

The results of the observational study [3] jointly with the results of the systematic mapping study [1] helped us in the *construction of MSECO-DEV*, an app development process that encompasses the developer’s activities within an MSECO, including recommendations and practices associated to their activities. This approach is presented in the next sections. In the Sections IV and V, we discuss the last two steps of the methodology we followed to build MSECO-DEV.

B. Definition of MSECO-DEV

In a MSECO, a keystone needs to support communication and coordination of external developers throughout processes’ phases, as well as to investigate the impacts of different practices in project planning, integration and development standards [7]. In MSECO-DEV, activities provide information and allow the development of an app aligned with keystone’s goals. The notation used to describe the MSECO-DEV is based on SPEM language¹. The artifact generated at the end of this process should be a publishable file packaged built from standards provided in the MSECO and must be supported by its platform. This artifact will be available in the app store, and it can be embedded in a user’s device.

For each activity, a *recommendation* consists of a guide to perform an activity to which it has been associated with. In turn, for each recommendation, exercises to achieve concrete results regarding the keystone’s goals were associated, labeled as *practices*. The practices are only associated with the activities prior to submission of an app to the store. In this section, we present a set of 8 recommendations (one per activity) and 14 practices which compose the MSECO-DEV process (Table 1). Once MSECO-DEV recommendations and practices were built, we evaluated them with developers of different MSECOS. Thus, we planned and conducted two surveys (Sections IV and V).

IV. FIRST SURVEY WITH EXPERIENCED DEVELOPERS

This section presents a survey planned and executed with the goal of **analyze** recommendations and practices that compose the MSECO-DEV **with the purpose of** characterizing **with respect to** their usefulness **from the point of view of** experienced developers **in the context of** app development activities in MSECO.

A. Survey’s Research Questions and Instrumentation

The research questions (RQs) and their related metrics are described in the Table 2. As instrumentation, we prepared a questionnaire to evaluate the relationship between practices and recommendations with the activities of the MSECO-DEV through the assertion: “*Is [recommendation/practice] related to the activity [activity] as in your routine?*”. 5-points Likert scale was used offering the following options: 1. *Totally Agree*; 2. *Partially Agree*; 3. *I do not know (neutral)*; 4. *Partially Disagree*; and 5. *Totally Disagree*.

¹ <http://www.omg.org/spec/SPEM/>

Table 1. Activities, Recommendations and Practices.

[Activity 1] to define the app idea (scope), taking into account the devices, and also APIs and SDKs used in the MSECO.
Recommendation: Consider how hardware and operating system software features (e.g. screen navigation control) can affect app development.
(P1) Use support materials when answering questions about the correct definition of the developer's idea;
(P2) Define scope to confirm the viability of the idea;
(P3) Analyze a set of apps as well as non-existing apps in the MSECO;
(P4) Analyze a set of successful apps within App Store;
(P5) Identify niche opportunities for app development, i.e., society needs;
(P6) Analyze non-successful apps within the app store.
[Activity 2] to prepare the app's marketing material in order to help developers to create user interface compatible with the platform.
Recommendation: Evolve developer's idea to prepare marketing material for an app; images and icons must follow standards;
(P7) Prepare apps based on support to generate marketing materials that can be used within the app store and in other information channels.
[Activity 3] to develop the app through tools provided by ecosystem. At the end of this activity, there is a binary.
Recommendation: Use design standards for the app development.
(P8) Use design patterns in app development;
(P9) Apply app store's acceptance criteria in the app development;
(P10) Use platform interface design guides to ensure the app identity;
(P11) Develop an app driven by user experience.
[Activity 4] to analyze binary against the quality criteria of the app store.
Recommendation: Generate a document with the results of the analysis.
(P12) Use tools to enable the verification of the binary;
(P13) Use binary validation tools;
[Activity 5] to analyze marketing package according to the guidelines.
Recommendation: Include images of app screens into the package.
(P14) Use a checklist to ensure the completeness of the generated marketing.
[Activity 6] to submit the binary and the marketing package to the developer central portal.
Recommendation: Be aware of submission rules that can be found in the support material.
[Activity 7] to monitor the app acceptance status in the store.
Recommendation: Monitor apps via e-mails or check the developer central portal.
[Activity 8] to monitor the reports published in the store.
Recommendation: Consider users' comments and reviews related to app evolution, and release updates in order to add new features.

Table 2. First Survey – Research Questions and Metrics.

RQ1. What are the recommendations to the app development activities within an MSECO?
Metric: the list of recommendations of MSECO-DEV process that are effectively useful in the app development activities (recommendations consolidated from the initial set, or added by participants of the study).
RQ2. What are the practices in the app development within an MSECO?
Metric: the list of practices of MSECO-DEV process that are effectively useful in the app development activities (practices consolidated from the initial set, or added by the participants of the study).

B. Participants' Profile

The distribution of these developers regarding their experience in app development is the following: 22 (individually), 8 (as a part of a team, startup, or only a team) and 10 (as a part of a team in industry). The participants were asked to inform in which MSECOs they work (or already worked). The results obtained for the three major MSECOs are: 40% (Windows Phone), 30% (Android), 13% (Other 'dead' Nokia's MSECOs), 10% (Symbian), and 7% (iOS). As mentioned, they could act in more than one MSECO.

C. Analysis of the Recommendations

Regarding the recommendations, Table 3 presents the number of responses for each option available in the questionnaire: total agreement (TA), partial agreement (PA), if the participant had no formal opinion (NO), partial disagreement (DP), or total disagreement (TD). In addition, there is a column in Table 3 labeled *Disagreement Level (DL)* to represent the percentage of negative responses (PD or TD). As highlighted in Table 3, three recommendations (38%) did not reach $DL = 0\%$. It means that five recommendations (62%) were confirmed by participants, that is, $DL = 0\%$. The recommendations with some level of disagreement are associated to Activities 2, 4 and 5.

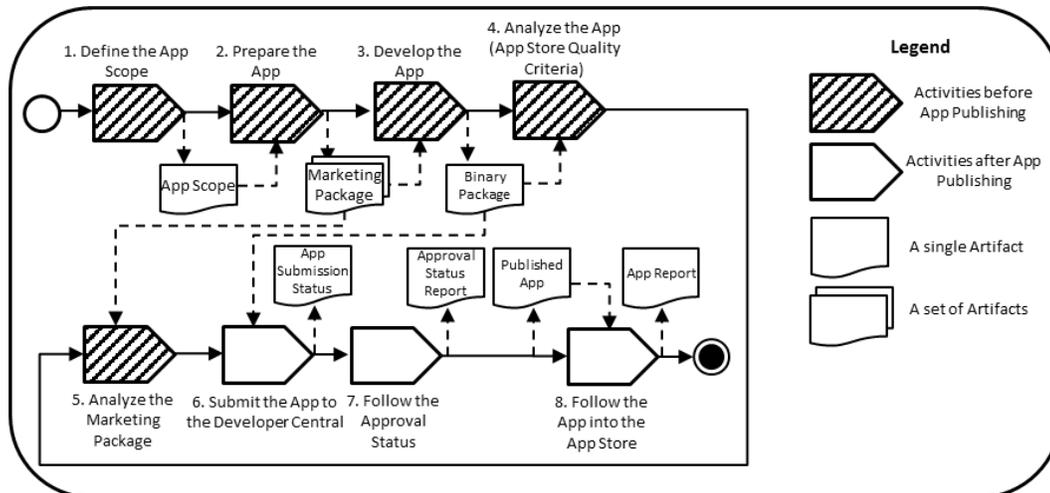


Figure 1. MSECO-DEV process.

The disagreement regarding the recommendation associated to Activity 2 might be caused because it is part of an initial activity in the process, according to a developer: “*It is a bit rash to think on how to sell something that you do not have yet; many things and ideas will possibly change along the development*”.

Table 3. Analysis of Practices and Recommendations.

Activity	Practice	TA	PA	NO	PD	TD	DL
1	Recommendation	29	4	0	0	0	0%
	P1	26	6	1	0	0	0%
	P2	26	6	1	0	0	0%
	P3	19	11	3	0	0	0%
	P4	19	11	2	1	0	3%
	P5	22	9	2	0	0	0%
2	Recommendation	18	12	0	3	0	9%
	P7	20	10	2	0	1	3%
3	Recommendation	16	13	4	0	0	0%
	P8	19	13	1	0	0	0%
	P9	28	4	1	0	0	0%
	P10	19	11	3	0	0	0%
4	Recommendation	20	8	2	2	1	9%
	P12	21	9	2	0	1	3%
	P13	19	9	5	0	0	0%
5	Recommendation	20	8	2	2	1	9%
	P14	24	8	1	0	0	0%
6	Recommendation	29	4	0	0	0	0%
7	Recommendation	29	2	2	0	0	0%
8	Recommendation	30	3	0	0	0	0%

In the Activity 4, there was a disagreement in the recommendation reported by a developer as follows: “*It is not necessary to generate a document summarizing the result of the analysis*”. In the recommendation associated with Activity 5, the disagreement was observed on the marketing package and images, since there is a need for preparation of promotional video and marketing strategies – and it is a complex task in the application development.

D. Analysis of the Practices

As highlighted in Table 3, four practices (29%) did not reach $DL = 0\%$. It means that 10 practices (71%) were confirmed by participants, that is, $DL = 0\%$. However, the participants have provided some suggestions for the adjustment of the 4 practices.

The Practice P4 related to the Activity 1 got DL greater than 0%. With regard to this practice, participants commented: “*The analysis of apps (successful or not) can be seen only as an indication, but should not be a fundamental item*”.

The Practice P7 related to Activity 2 obtained some level of disagreement. The disagreement seemed to be associated with the fact that, perhaps, it is not the best moment to think about

marketing (or maybe the developer is not the best role to prepare strategies and marketing).

Regarding the Activity 3, the Practice P11 had some level of disagreement. It was reported that “*the usability concern is a requirement. The use of templates – wireframes, prototypes, and mockups, for example – also serves for this purpose*”.

In the practices associated with Activity 4, a participant disagreed with the Practice P12 because he/she “*understands the operation of binary validation tools as part of the app’s validation and also as part of the analysis of the development feasibility*”.

E. New Suggested Practices

During the study, participants suggested three new practices, described as **NewP**<Sequence Number> – **Activity** <number>, , which was included in the MSECO-DEV (Table 4).

Table 4. First Survey – New Suggests Practices.

NewP1 – Activity 1: Pay attention to the gaps of similar apps, and to the comments of frustrated users, in order to be able to meet the ecosystem’s expectations.
NewP2 – Activity 1: Evaluate the effort and complexity of using an app, if it uses or requires third-party APIs.
NewP3 – Activity 2: Prepare the app to offer a consistent user experience, since users will always recommend others with the apps they like.

F. Threats to Validity

Conclusion validity: accomplished through simple demonstration of presence (or not) of recommendations and practices stated in the MSECO-DEV.

Internal validity: for this study, we proposed to select developers who work in the main MSECOs. Thus, we assumed that they are representative for the population of MSECO developers.

Construct validity: the study is characterized by applicability analysis of the recommendations and practices associated with the MSECO-DEV activities with respect to the current activities required by the app development in MSECO.

External validity: as mentioned in Internal Validity, the participants act in the main MSECOs. However, new studies could be performed with more developers.

V. SECOND SURVEY WITH NOVICE DEVELOPERS

The goal of this second survey was to **analyze** a subset of practices and activities that compose the MSECO-DEV **with the purpose of characterizing with respect to** their usefulness and applicability **from the point of view of novice developers in the context of** app development activities in MSECO.

A. Research Questions and Instrumentation

The RQs and related metrics to help us to answer them are presented in Table 5.

Table 5. Second Survey – Research Questions and Metrics.

RQ1. Are the activities that compose MSECO-DEV applied by developers before the submission of an app to the app store?
Metric: the list of MSECO-DEV activities applied in the app development.
RQ2. Are the practices related to the activities that compose MSECO-DEV useful and applicable by developers before the submission of an app to the app store?
Metric: the list of MSECO-DEV practices that are useful and applied in app development.

As the first survey, a questionnaire was prepared to evaluate the usefulness and applicability of the activities and practices, and the existing relations among them.

B. Participants' Profile

We invited novice developers who are undergraduate or graduate students attending a Mobile Applications Engineering course. During the course, developers had contact with professionals from academia and industry covering topics necessary for the development of apps, such as monetization, design, testing, user experience, tools etc. At the end of the course, they developed an app applying the concepts they learned.

In total, 32 developers participated in this study. When were asked about which MSECOS they work or worked, the results were: 32 (100%) work/worked in the Android, 3 (9%) work/worked in the iOS, and 5 (16%) work/worked in the Windows Phone. Participants did not use MSECO-DEV to develop the app during the course.

C. Analysis of Activities

As mentioned in the purpose of the study, the participants have reviewed the activities that compose the pre-publication stage. In other words, the developers did not submit their mobile apps and they did not have the experience in evaluating the remainder activities. The following activities presented in Section III.B were evaluated: 1 – Define the App Scope, 2 – Prepare the App, 3 – Develop the App, 4 – Analyze the App (App Store Quality Criteria), and 5 – Analyze the Marketing Package.

Each participant was asked to inform the activities they performed during the development of their projects (FP – *Fully Performed*, or PP – *Partially Performed*), or whether he/she did not perform some of them (NP – *Not Performed*). Table 6 shows the number of responses for each activity. From this data, it was possible to observe two main results. 19 developers (59%) did not analyze the mobile applications against the app store criteria after the development (Activity 4). Moreover, 24 developers (75%) did not analyze the marketing materials prepared for the app (Activity 5).

Regarding the Activity 4, the main critical comments were related to: 1) the app is still under development; 2) the lack of experience; 3) there was no specific time to perform a full analysis of the quality criteria; 4) another team should perform

this activity; 5) there was no interest in making the app available on the app store; and 6) the lack of knowledge of the activity.

Regarding Activity 5, the developers listed as reasons to not execute it: 1) the lack of support and technical material; 2) the lack of time; 3) the development was not completed yet; and 4) no focus on providing the mobile application to the app store.

D. Analysis of the Practices

Concerning the practices, participants answered the questions selecting options from a 4-scale of usefulness/applicability: a) useful and already applied (U/A); b) useful and not applied yet (U/NA); c) not useful and already applied (NU/A); d) not useful and not applied yet (NU/NA). Table 6 presents the participants' responses to each practice. After analyzing specifically the practices in which there is a great number of answers *useful and not applied yet*, we identified six practices as highlighted in Table 6 – all included in the following analysis.

Table 6. Usefulness and Applicability of Practices.

Activity	FP	PP	NP	Practice	U/A	U/NA	NU/A	NU/NA
1	15	17	0	P1	25	6	1	0
				P2	31	1	0	0
				P3	27	4	0	1
				P4	22	10	0	0
				P5	19	13	0	0
				P6	11	18	1	2
				NewP1	19	13	0	0
2	14	15	3	NewP2	15	13	0	4
				P7	5	22	2	3
				NewP3	21	11	0	0
3	31	1	0	P8	23	9	0	0
				P9	7	25	0	0
				P10	22	10	0	0
				P11	27	5	0	0
4	5	8	19	P12	6	25	0	1
				P13	3	28	0	1
5	4	4	24	P14	2	30	0	0

In the Activity 1, P6 was not applied by 18 (56%) participants. A participant commented that: “*Our team should consider enough time to check the feasibility of our app*”.

About P7, even considering this practice useful, 22 participants (69%) did not implement it in the app development. About this practice, a participant gave the following feedback: “*The ‘first impression’ of the user is always important. An app gets ‘fame’ with the comments from other users*”.

For P9, associated with Activity 3, 25 participants (78%) did not apply it even considering it useful. The feedback that summarizes the reason the practice has not been applied is: “*I knew these criteria but I did not know where to find them*”.

The two practices of Activity 4, P12 and P13, have been evaluated respectively as useful, but they were not applied by 25 (78%) and 28 (88%) participants, respectively. In respect to P12, a participant commented that: “*I did not know where the tool*

was and even how to use it in practice”. About P13, an answer was: “As we had no idea on how to submit the app to the store, we did not perform the tests as frequently as we should do”.

Finally, in the Activity 5 (before the submission of the app to the store), P14 was evaluated by 30 participants (94%) as useful, but they never applied it. A feedback from this evaluation is the following: “I did not analyze the lack of knowledge of the marketing guides for my app”.

Even considered useful, this subset of practices should be applied again after an adjustment in the sequence in which they appear. Therefore, those practices should appear during the development, or prior to development. The practices can be adjusted in activities that define the scope of the app and prepare initial marketing packages for app.

E. Threats to Validity

Conclusion validity: accomplished through simple demonstration of presence (or not) of activities and practices of the MSECO-DEV.

Internal validity: for this study, we proposed to select developers who participate in MSECOS. Thus, we assumed that they are representative for the population.

Constructo validity: the study is characterized by analysis of the activities’ sequence and practices, as well as the description of practices and activities and the association among them of the development process with respect to activities and practices necessary for the development of apps in MSECO.

External validity: Participants can be considered as a sample of the population once they received specific training sessions for the development of apps.

VI. CONCLUSION AND FUTURE WORK

In the new business model raised from mobile platforms, even with the keystone’s investment in the development support, few practices and processes have been defined to support the developer’s work, as confirmed in [1]. Another point is that the keystone needs to expand the MSECO through the growing number of apps of good quality (star rating of apps) to attract more users. Due to this fact, the external developers – outside the keystone organization – need to develop apps that help to reach the keystone’s goals.

In this scenario, we presented a process named MSECO-DEV that aims to support the developer in the construction of an app in the MSECO context. As an important contribution of this work, we can mention the set of activities, recommendations, and practices of the MSECO-DEV, that were defined and evaluated based on experimental studies.

As future work, we propose: (a) to perform case studies with at least two MSECOS in order to apply MSECO-DEV; (b) investigate other processes in MSECO, such as orchestration, which is focused on the keystone; (c) analyze the sources of Developer eXperience (DX) involved in app development process adopted by MSECOS; and (d) examine how an MSECO’s processes impact the health of the ecosystem, i.e., the ability of a MSECO to survive the disruptions (e.g., developers escape) and remain productive.

ACKNOWLEDGMENT

The authors would like to thank FAPEAM, CNPq, CAPES (Proc. no. BEX 0204/14-5), and INDT for the financial support.

REFERENCES

- [1] A. Fontão, R. P. Santos, A. C. Dias-Neto. 2015. Mobile Software Ecosystem (MSECO): A Systematic Mapping Study. In *Proceedings of the 39th Annual International Computers, Software & Applications Conference (COMPSAC)*, Taichung, pp. 653-658.
- [2] A. Fontão, R. P. Santos, A. C. Dias-Neto. 2015. Research Opportunities for Mobile Software Ecosystems. In *Proceedings of the 9th Workshop on Distributed Software Development, Software Ecosystems and Systems-of-Systems (WDES)*, Belo Horizonte, pp. 97-98.
- [3] A. Fontão, B. Bonifácio, A. Dias-Neto, A. Bezerra, R. P. Santos. 2014. MSECO Skill: Construção de Competências de Desenvolvedores em Ecosystemas de Software Móvel. In *Proceedings of the 17th Iberoamerican Conference on Software Engineering (CIbSE)*, Pucón, pp. 81-94.
- [4] L. Corral, A. Sillitti, G. Succi. 2013. Software development processes for mobile systems: Is agile really taking over the business? In *Proceedings of the 1st International Workshop on the Engineering of Mobile-Enabled Systems (MOBS)*, San Francisco, pp. 19-24.
- [5] S. Lim, P. Bentley, N. Kanakam, F. Ishikawa, S. Honiden. 2015. Investigating Country Differences in Mobile App User Behavior and Challenges for Software Engineering. *IEEE Transactions on Software Engineering* 41(1):40-64.
- [6] F. Fagerholm, J. Münch. 2012. Developer experience: Concept and definition. In *Proceeding of the International Conference on Software and System Process (ICSSP)*, Zurich, pp. 73-77.
- [7] I. Wasserman. 2010. Software engineering issues for mobile application development. In *Proceedings of the FSE/SDP Workshop on Future of Software Engineering Research (FoSER '10)*, Santa Fe, pp. 397-400.

Layered Implementation View of a SOA Based Electronic Health Record

Josimar S. Lima², Joyce M. S. França¹, Jislane S. S. Menezes², Adicinéia A. Oliveira² and Michel S. Soares²

¹Faculty of Computing, Federal University of Uberlândia, Uberlândia, Brazil

²Department of Computing, Federal University of Sergipe, São Cristóvão, Brazil
{osimar.lima2007, joycefranca, jislanesds, adicineia, mics.soares}@gmail.com

Abstract

Interoperability between legacy systems has crucial importance for integration of many distributed systems. Within health information systems, the need for interoperability between legacy systems is well-known, as these are heterogeneous systems which have a long lifespan. An interesting example of health information systems is the Electronic Health Record (EHR), a virtual record of every health-related event, including hospital admission, general practitioner visit, exams, and allergies experienced by individuals over their lifespan from in utero to death. A great challenge for implementing EHR applications is that these systems often have to integrate large amount of data gathered from legacy systems into a single application. This paper describes in detail the implementation view of a software architecture based on web services for developing an EHR application in a public hospital. The architecture has been defined with a layered style for implementation and on services identified within current legacy systems. Detailed explanation on the software development, including architectural elements and services catalog are described as well.

Keywords-Interoperability, Service Oriented Architecture, Layered Implementation, Health Information Systems.

I. Introduction

Health information systems are becoming common in health institutions not only for management purposes, but also to improve health care. Complexity of developing

these software systems is well-known in the literature for many reasons, including heterogeneity of data and issues with interoperability between legacy systems [1], and design and architecture problems [2]. In addition, health applications are sensitive to important non-functional requirements, such as performance, security and safety of data [3] [4]. Therefore, developing and managing health information systems is a hard task. New functionalities arises constantly due to new ideas or government laws, and they have to be implemented and integrated to other legacy systems.

An Electronic Health Record (EHR) is a virtual record of every health-related event (e.g., hospital admission, general practitioner visit, exams, allergies) experienced by individuals over their lifespan from in utero to death [5]. EHR systems can be considered complex health information systems because they are used for many purposes, including recording data for decades and providing information gathered from data stored in many legacy software systems, developed using different technologies, databases, platforms and programming languages. EHRs normally have to integrate large amount of data gathered from legacy systems in a single application.

Benefits of an EHR system are better patient care, improved and faster communication with other systems, including external systems (government, health insurance, banking), improved sharing of data between health professionals, improved decision making, among others [6] [7]. However, these are complex systems to develop, maintain and operate. Failures during development or execution of EHR systems have been documented in the literature. Among the reasons for failures during development, a systematic literature review [5] mentioned a number of issues, including low level of user involvement, the need of redesigning work practices, and redesign of the record

format.

More specifically, the core subject of this paper is interoperability between legacy systems to implement a new EHR system. This problem has been addressed before in many ways [8], including by means of Enterprise Application Integration and Service Oriented Architectures [9]. For instance, the authors of paper [10] presented a framework with focus on defining services based on a Model-Driven Architecture approach supported by a SOA meta-model. In [11], the authors present a solution to interoperability in healthcare based on a middleware software architecture used in enterprise solutions. The general architecture is described, but implementation view and further development details are not explained. Authors of paper [12] describe an attempt to use SoaML with the purpose of modeling a real problem of system interoperability in health information systems. Integration of data in Electronic Health Information Systems is discussed in [13], in which, according to the authors, data interoperability in healthcare is, at present, largely an unreachd goal. Therefore, the authors suggest the adoption of a standardized healthcare terminology, as well as the connection of legacy systems to the health network as ways of achieving complete interoperability of Health Information Systems.

Developing an EHR considering interoperability constraints between legacy software systems remains an interesting research problem. In this research, the issue of interoperability is considered by designing a Service Oriented Architecture (SOA) in order to integrate legacy systems by means of web services. An EHR application is developed as a case study. The layered implementation view of the architecture is presented, as well as detailed explanation on the software development, including architectural elements and services catalogue.

II. Layered Architectural Ambient

Developing complex software systems is considered easier when a proper software architecture is proposed to describe major elements, global structure and design decisions [14]. Architectural environment in this paper is based on SOA Reference Model (SOA RM) and on SOA Reference Architecture (SOA RA).

SOA RM defined the vocabulary of SOA elements and its context relationships. SOA RM is created to semantically establish services definitions, contract description, service propagation, data model and service contracts [15]. SOA RM proposes a local basis on which reference architectures, software structure and implementation can be developed.

SOA RA is composed of a variety of models and specifications which define a logical platform for implementation. SOA RA combines SOA RM concepts with common

IT architectures by using models and visualizations of common architectural domains.

Some commercial vendors of SOA solutions propose well-defined reference architecture templates for developing SOA applications. Our proposal in this research is to use only open source software. Our reference architecture has no close connection to any platform. SOA reference architecture used in this research is the one proposed by The Open Group [16], a solution often applied to various projects since 2002. The Open Group reference architecture provides an instrument to create or evaluate an architecture in terms of layers, components (building blocks), and roles to be considered in such a way to assure return of investment in technology and that the objectives are achieved.

Fig. 1 depicts the architectural ambient proposed in this research based on The Open Group SOA RA. The proposed architecture is composed of five horizontal layers (functional) and four vertical layers (non-functional). Functional layers are described as follows.

- Operational Systems provide basic infrastructure to provide SOA functions. Besides, promotes integration with legacy systems, data bases, and so on, and allows services execution.
- Software components which implement the services. These components connect (bind) the Service Contract with its implementation, and provide loose coupling between consumer and implementation.
- Services work as a Service container in which we can find all Service Contracts, including Service Tasks, Entity, Information, and so on.
- Business Process, layer responsible for composition and orchestration of services. It supports long process, and executes tasks (sequential or parallel) according to policies, business rules, and constraints.
- Consumer Interfaces, layer which deals with communication with users, giving support to different channels between users and applications. It also provides communication between applications and loosely coupling between consumer and implementation.

Non-Functional layers are described as follows.

- Integration, which allows mediation, transformation, routing and transport. Services are exposed only through this layer, which centralizes business rules and provides loosely coupling between provider and consumer. This layer is generally supported by an ESB-Enterprise Service Bus.
- Quality of Service (QoS), which captures and monitors operational metrics, assuring reliability, availability, control, scalability and security.
- Information, including data architecture, data structures (XML-Schemas) and data protocols.
- Governance, which defines SOA objectives and as-

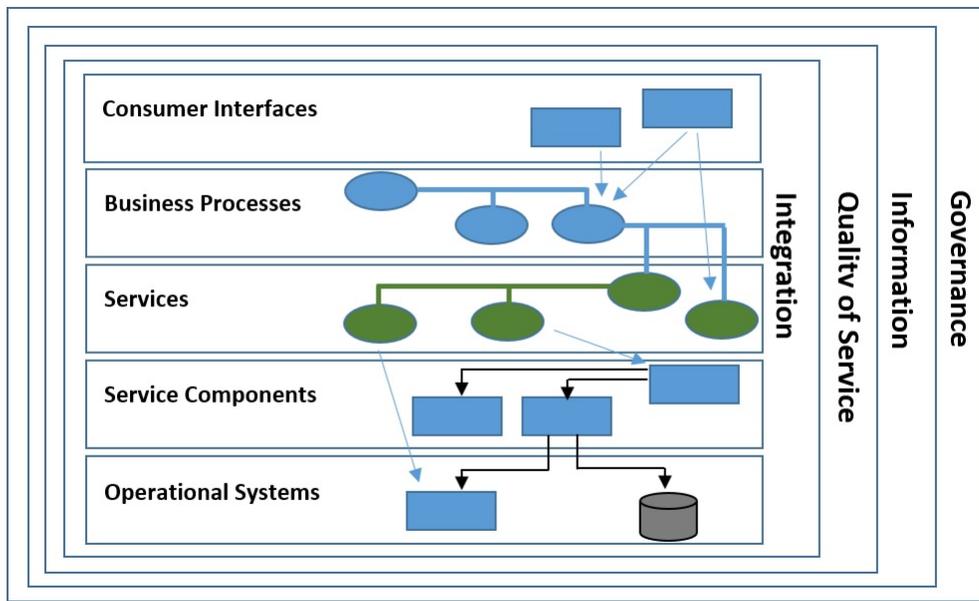


Fig. 1. Proposed Architectural Environment

sures conformity between policies and processes. It also defines solution portfolio. Governance is applied to all layers.

III. Architectural Elements

TABLE I. Architectural Elements

Analysis	Design	Implementation
Data	Relational Data Base	PostgreSQL
Systems Integration	XML Interface	Web Services
Web Services Integration	ESB	Mule ESB
Distribution layer	Object-Relational Mapping	Hibernate
Front-End	User interface communication	JSF, AJAX, PrimeFaces
Exception Handling	Exception Layer	Java
Deploy	IDE configuration for Deploy	Apache Maven
Log	Log resources implementation	Log4J

This section describes the architectural elements (Table I) that will compose the architecture, as described in the architectural ambient.

Architectural elements represent fundamental technical concepts standardized throughout the solution. They are refined during project development into three categories:

analysis, design and implementation. These categories reflect the state of the architectural element in time.

Each state changes according to successive levels of detail discovered during requirements refinement. For instance, during initial analysis phases, data to be used in the EHR application are identified. Then, the correspondent design element is a relational data base. Further, correspondent implementation elements are tables implemented in a Relational Data Base, PostgreSQL in this project.

IV. Architecture Implementation View

Proposed architecture in this paper use concepts from the reference architecture depicted in Fig. 1 and architectural elements presented in Table I.

The Implementation View of the architecture is composed by Mule ESB 3.6 as Enterprise Service Bus, services catalogue, and flow of services. The view is based on the layered style, described by a UML Package diagram as depicted in Fig. 2.

Fig. 2 depicts how packages are organized. Files related to flow of services generated by mule ESB as well as web services source code are organized in package App.

Figs. 3 and 4 depict the flow of services responsible by the bus management.

Input bus is the single entry point of a message from the services bus. In this initial stage, important tasks, such as security, input log, auditing, transformation, formatting and validation are executed. After these functions, the message flow is routed for the specific service asked by

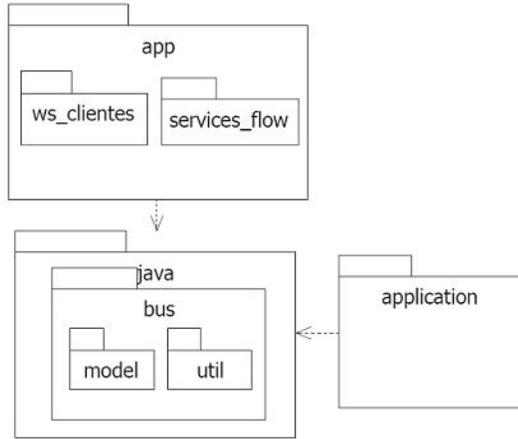


Fig. 2. ESB Package

the consumer. As depicted in Fig. 3, Mule receives objects through HTTP and executes all necessary transformations.

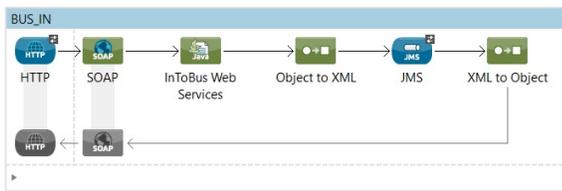


Fig. 3. Bus In

Output bus is the single output point in the services bus, being responsible for routing responses from the flow of services to the consumers. Some necessary tasks to finalize the message flow in the service are executed here, including log registry, data formatting, data transformation, validations, among others. Output bus implemented in this application is presented in Fig. 4, which makes it clear that the output bus does the inverse operation of the input bus.

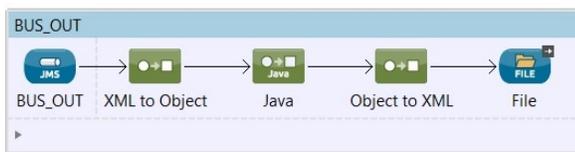


Fig. 4. Bus Out

Package Java has bus as subpackage, which has bus util and model as subpackages. Subpackage bus has source code to manipulate services flow as depicted in Figs. 3 and 4. Package model has source code related to entities that will be retrieved by client web services. Package util has source code used to manage services flow. Package

applications is the package to store source code related to all applications developed based on this implementation architecture.

V. Case Study

Our proposed architecture has been considered for development of an EHR application in a public hospital. Development of the application is based on the MVC pattern, as depicted in Fig. 5, and the architectural elements presented in Table I.

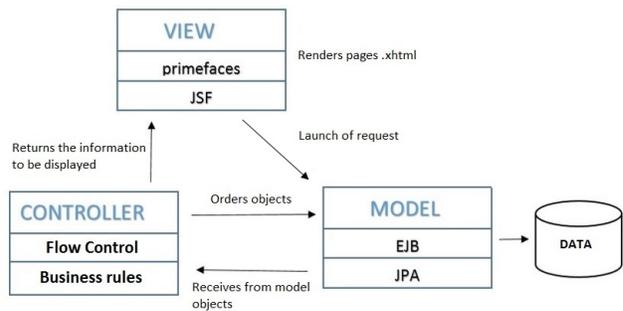


Fig. 5. MVC model for application implementation

The developed EHR application involves a number of legacy systems. Each one provides important patient information that will be consumed by the EHR. From the legacy systems which are integrated to design the EHR, Services are identified and then modeled by using the SoaML modeling language [17].

TABLE II. Service Catalogue

ID	DESCRIPTION	PROVIDER
1	Data for Login	AGHU
2	Create medical appointment	ACONE
3	Validate health card	CADWEB
4	Get patient's data	AGHU
5	List of appointments	MEDLYNX
6	List of exams	MEDLYNX
7	List of medical hospitalization	IMHOTEP
8	List of procedures	AGHU

Patient registration is performed by ACONE, which maintains also appointments schedule. CADWEB, a national health system contains registration of country citizens. Each registered citizen in CADWEB has a unique health card number, a national identifier that allows free access to public hospitals, exams, appointments, medicines and surgeries. AGHU is the hospital's main legacy system that contains important information about patients. One module of AGHU is responsible for maintaining patient personal data such as name, address, identity documents,

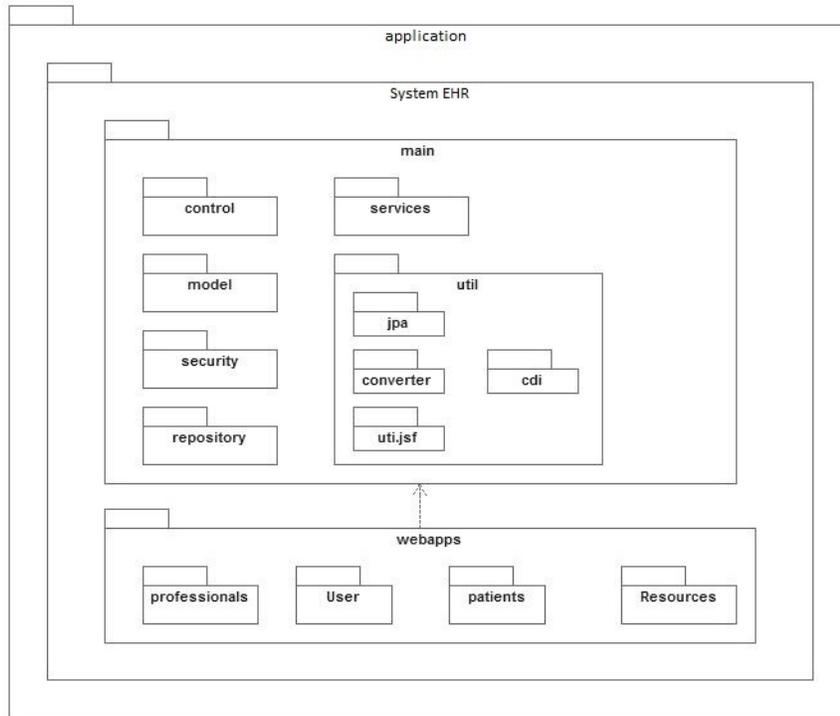


Fig. 6. Application components

and health card number. The other AGHU module controls patient exams, including storing all exams to which patients were submitted, exam results and a list of approved and pending exams. There are yet other modules that are responsible for maintaining history of attendances, hospitalization, surgery, medicines and medical procedures. Medlynx is a legacy system used to manage medical appointments and exams. Imhotep is a legacy system responsible to manage financial data and all information about products at the hospital, including medicines and other health related materials.

After analyzing all legacy systems we obtained a list of services. Services catalogue is the local where all identified services are registered. These services (Table II) can be used by the whole organization in their process behavior with the purpose of achieving a service oriented integration.

Java is the programming language used for implementation, together with frameworks Java Server Faces and Primefaces. Postgre is the relational database, together with Hibernate as JPA framework. We have also used Apache Maven for dependency injection, JBoss as application server, and Mule ESB as Enterprise Service Bus.

Fig. 6 depicts the organization of the implementation view applied for development of the EHR application. This picture details package application, as previously presented

in Fig. 2.

VI. Conclusions

Developing new software systems from scratch is frequently not possible, as there are many legacy systems with stored data that need to be maintained. With this in mind, the SOA paradigm is useful to integrate legacy systems by means of web services. For this reason, considering that having a well-defined software architecture is crucial for success of complex systems, this paper proposes and describes the layered implementation view of a software architecture to develop EHR systems based on the SOA paradigm to integrate legacy systems. This view of the architecture has been used to develop an EHR in a public hospital.

The complexity of developing health information systems is well-known in the literature for many reasons, including heterogeneity of data and issues when defining the system architecture. In addition, within health information systems, the need for interoperability between legacy systems has been described in the literature, as these are normally heterogeneous systems which have a long lifespan.

Complexity of Electronic Health Record (EHR) development is addressed in this paper by proposing a layered

style for implementation. The choice for interoperability in this paper is to identify services from legacy systems, providing a services catalogue, and then integrating them into an EHR application. This choice was taken as the SOA paradigm is useful to integrate legacy systems by means of web services. Future research will focus on exploring further other views of the architecture for developing the EHR application.

Acknowledgment

The authors would like to thank the Brazilian research agency CNPq (grant 445500/2014-0).

References

- [1] T. Greenhalgh, K. Stramer, T. Bratan, E. Byrne, Y. Mohammad, and J. Russell, "Introduction of Shared Electronic Records: Multi-site Case Study using Diffusion of Innovation Theory," *BMJ*, vol. 337, no. 7677, pp. 1040–1044, 2008.
- [2] P. Carayon, P. Smith, A. S. Hundt, V. Kuruchittham, and Q. Li, "Implementation of an Electronic Health Records System in a Small Clinic: the Viewpoint of Clinic Staff," *Behaviour & IT*, vol. 28, no. 1, pp. 5–20, 2009.
- [3] T. Greenhalgh, S. Hinder, K. Stramer, T. Bratan, and J. Russell, "Adoption, Non-adoption, and Abandonment of a Personal Electronic Health Record: Case Study of HealthSpace," *BMJ*, vol. 341, no. 7782, 2010.
- [4] L. Boyer, J. Samuelian, M. Fieschi, and C. Lancon, "Implementing Electronic Medical Records in a Psychiatric Hospital: a Qualitative Study," *Int. J. of Psychiatry Clinical Practice*, vol. 14, no. 3, pp. 223–227, 2010.
- [5] L. Nguyen, E. Bellucci, and L. T. Nguyen, "Electronic Health Records Implementation: An Evaluation of Information System Impact and Contingency Factors," *Int. J. of Medical Informatics*, vol. 83, no. 11, pp. 779–796, 2014.
- [6] S. Zimeras and A. N. Kastania, "Statistical Models for EHR Security in Web Healthcare Information Systems," *Certification and Security in Health-Related Web Applications: Concepts and Solutions: Concepts and Solutions*, p. 146, 2010.
- [7] A. Sheikh, A. Jha, K. Cresswell, F. Greaves, and D. W. Bates, "Adoption of Electronic Health Records in UK Hospitals: Lessons from the USA," *The Lancet*, vol. 384, no. 9937, pp. 8–9, 2014.
- [8] L. D. Xu, "Enterprise Systems: State-of-the-Art and Future Trends," *IEEE Transactions on Industrial Informatics*, vol. 7, no. 4, pp. 630–640, Nov 2011.
- [9] M. S. Soares and J. M. S. França, "Characterization of the Application of Service-Oriented Design Principles in Practice: A Systematic Literature Review," *Journal of Software*, vol. 11, no. 4, pp. 403–417, 2016.
- [10] S. Alahmari, E. Zaluska, and D. D. Roure, "A Service Identification Framework for Legacy System Migration into SOA," in *IEEE Int. Conf. on Services Computing*, July 2010, pp. 614–617.
- [11] A. Ryan and P. W. Eklund, "A Framework for Semantic Interoperability in Healthcare: A Service Oriented Architecture based on Health Informatics Standards," in *Proc. of MIE2008, The XXIst Int. Congress of the European Federation for Medical Informatics*, 2008, pp. 759–764.
- [12] F. G. Silva, J. S. S. de Menezes, J. de S. Lima, J. M. S. França, R. P. C. do Nascimento, and M. S. Soares, "An Experience of using SoaML for Modeling a Service-Oriented Architecture for Health Information Systems," in *Proc. of the 17th Intern. Conf. on Enterprise Information Systems*, 2015, pp. 322–327.
- [13] O. Iroju, A. Soriyan, I. Gambo, and J. Olaleke, "Interoperability in Healthcare: Benefits, Challenges and Resolutions," *Int. J. of Innovation and Applied Studies*, vol. 3, no. 1, pp. 262–270, 2013.
- [14] G. Booch, "The Economics of Architecture-First," *IEEE Software*, vol. 24, no. 5, pp. 18–20, 2007.
- [15] OASIS. (2006) Reference Model for Service Oriented Architecture.
- [16] O. Group, "SOA Reference Architecture," Tech. Rep., 2011.
- [17] OMG, "Service Oriented Architecture Modeling Language (SoaML) Specification," Tech. Rep., 2012.

An Ontology-driven Adaptive System for the Patient Treatment Management

Emna Mezghani

LAAS-CNRS, Université de Toulouse, CNRS, INSA, Toulouse,
France

Luxembourg Institute of Science and Technology, 5, Avenue des
Hauts-Fourneaux, L-4362, Esch/Alzette, Luxembourg

Marcos Da Silveira, Cédric Pruski

Luxembourg Institute of Science and Technology, 5, Avenue des
Hauts-Fourneaux, L-4362, Esch/Alzette, Luxembourg

Ernesto Exposito and Khalil Drira

LAAS-CNRS, Université de Toulouse, CNRS, INSA, Toulouse, France

Abstract— Advances in the Web and healthcare data capture technologies have far-reaching benefits for the development of new clinical decision support systems that accelerate decision-making and generate personalized treatments. However, the diversity of healthcare data formats, the lack of computer interpretable representation of medical interventions, and the distribution of reliable medical knowledge sources constitute important barriers to better support the medical decision process. To deal with these issues, we propose the Treatment Plan Ontology (TPO) that formalizes medical interventions, and allows medical systems sharing and reasoning over them. This knowledge together with the acquired patient data are then reused by the autonomic processes that we have developed in order to timely detect anomalies and support the physicians in personalizing the patient treatment at the right time. We demonstrate the system efficiency through a use case for managing hyperglycemia in type 2 diabetes.

Keywords- *Treatment Adaptation; Autonomic Computing; Semantic Web; Knowledge Representation*

I. INTRODUCTION

Advances in healthcare technologies including mobile systems, medical sensors and wearable computing foster the generation of real time data. Leveraging these technologies to automate the integration of the patient data with the medical knowledge paves the way for developing smart clinical decision support systems (CDSS).

The efforts of Medical and Informatics communities to formalize medical knowledge from clinical guideline resulted into several computer-interpretable guidelines (CIGs) described in different languages to elucidate and facilitate the decision process [1]. Recently, considerable efforts have been invested to bring the medical knowledge onto the Web using Semantic Web technologies [2] in order to integrate different sources, to update, to share and reuse this knowledge. Noticeably, drug-drug interactions, drug-food interactions and adverse drug reactions are being published and accessible as linked data [3, 4]. Integrating these sources with the medical interventions described by medical experts is crucial for more fine-grained decisions. Nevertheless, there is a lack of flexible representation of medical interventions that underpin the treatment adaptation and the dynamic patient context changes. Moreover, the acquisition of such knowledge is challenging

since it requires a minimum of IT expertise to annotate the medical interventions and update decision rules.

Consequently, we proposed an ontology-driven adaptive system that provides a smart management of the patient treatment. We defined a *Treatment Plan Ontology (TPO)* that will be used for annotating medical intervention and their associated rules. Semantic Web technologies were adopted to develop a collaborative user-friendly environment that can hide the complexity of the formalism used by the system. Moreover, we developed reusable autonomic processes that exploit the annotated knowledge to automatically detect anomalies and assist the physicians in taking the right decision at the right time.

The remainder of this paper is organized as follows. Section 2 details the proposed adaptive system and delineates the Treatment Plan Ontology. Section 3 highlights the efficiency of our adaptive system through a diabetes use case. Section 4 discusses existing work dealing with knowledge-based systems for treatment adaptation. Finally, Section 5 concludes this paper and gives some directions for future work.

II. PATIENT TREATMENT ADAPTATION

In this section, we present an overview of our adaptive system and the *Treatment Plan Ontology* which is the foremost element for automating the decision process.

A. An Overview of Treatment Adaptation

We adopted the autonomic computing paradigm [5] to adapt and manage the patient's treatment [6] in order to accelerate the decision-making and avoid the patient health complications. Originally inspired from the human autonomic nervous system, the autonomic computing has been proposed to enable self-managed properties through implementing the *MAPE-K* pattern (Monitoring, Analysis, Plan, Execution and Knowledge).

In our work, the *Monitoring* process is responsible of collecting patient data coming from medical sensors and wearable devices, and/or measured by health professionals. Once the data is collected, the *Analysis* process is triggered to identify patient health deterioration. It may implement detection rules, or predictive algorithms such as machine learning. According to the severity of the detected (or predicted) anomaly, an alert will be sent to the *Plan* process

that automatically searches and filters the appropriate treatments based on the patient medical conditions, the interventions contraindications, the medical interactions and the drug side effects. Finally, the personalized recommendations are sent to the appropriate physician for approval or modification. The aforementioned processes operate on the *Knowledge* which is the fundamental element that enables the coordination of the MAPE processes for the smart management of the patient treatment.

In this paper, we delineate the knowledge component and we propose the *Treatment Plan Ontology (TPO)* which semantically formalizes the medical interventions including their conditions and side effects, and represents some concepts pertaining to the patient context in order to assist the physicians

in making personalized decisions that meet the patient profile and preferences.

B. Treatment Plan Ontology

We followed a collaborative methodology [7] for formalizing the medical knowledge, based on discussions with medical experts. The output is a flexible TPO schema that builds the bridge between the artificial intelligence planning and the semantic representation of medical knowledge in order to grasp the computational intelligence. Our TPO semantically represents the medical interventions as actions characterized through preconditions and side effects in order to reach the patient objective. The main objective is to provide a generic model, which is the basis for reasoning, to automate the decision making according to the identified problems.

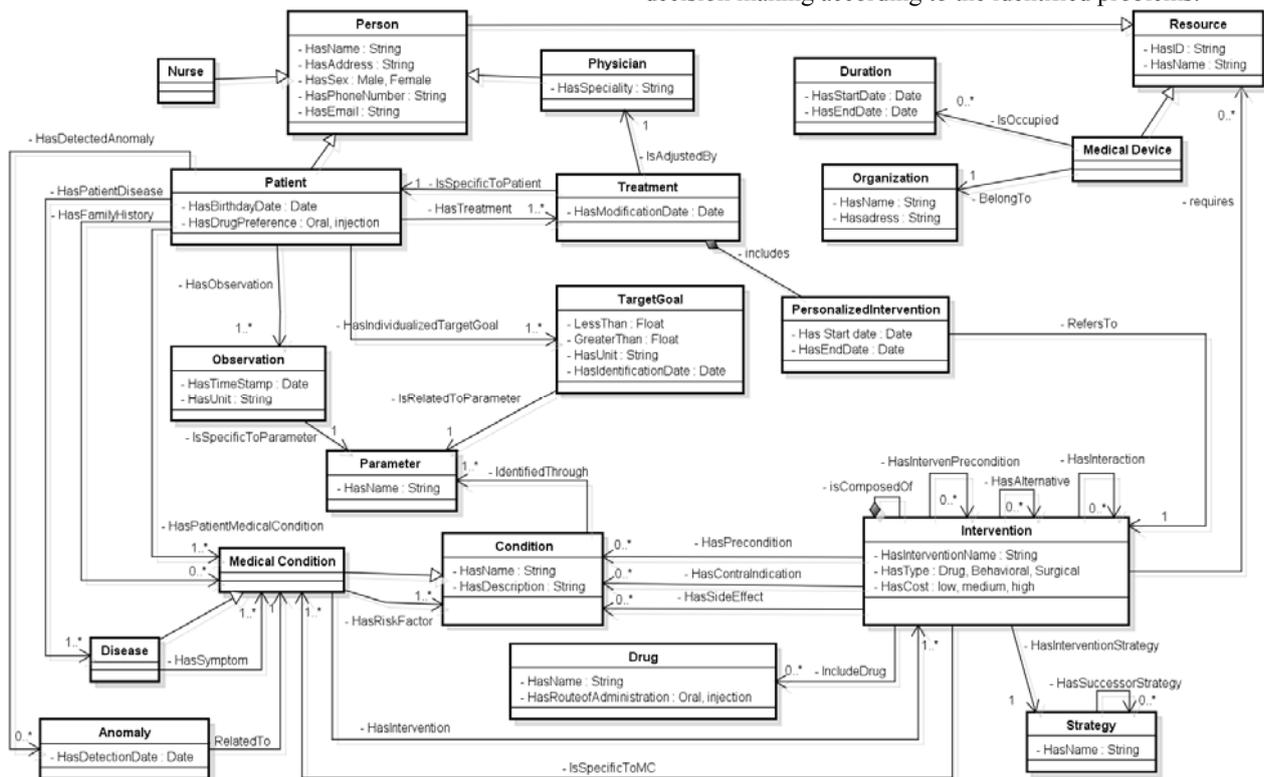


Figure 1. Depiction of the main classes and relationships with their cardinalities in the proposed Treatment Plan Ontology

Figure 1 portrays the main classes and relationships of our TPO that represents a formalization of a problem-solving system for chronic disease management. We introduced in TPO the medical condition class which is defined by the Segen's Medical Dictionary as “A disease, illness or injury; any physiologic, mental or psychological condition or disorder... A biological or psychological state which is within the range of normal human variation is not a medical condition”. Each condition is identified through measurable parameters, for instance the HbA1c identifies a high or low blood sugar medical condition. We consider that a disease has symptoms which are also medical conditions. Idem, each medical condition has risk factors which are defined as conditions that could make a person more likely to develop a disease or to amplify the symptoms of an existing disease. For example, both obesity and hypertension are risk factors of the type 2

diabetes. These medical conditions can be minimized and/or stabilized through medical interventions that help achieving the patient goal. An intervention can be behavioral, surgical or drug-based (characterized by its route of administration). To automate the selection of a treatment, our TPO associates each medical intervention with the appropriate disease strategy, and encodes the right strategy selection through the *HasSuccessorStrategy* property. For example, in type 2 diabetes, four strategies are identified: the “monotherapy”, the “dual combination”, the “triple combination” and the “complex insulin” [8]. If a patient who is treated with a “monotherapy” strategy presents an increased blood sugar, it is recommended to prescribe a “dual combination” strategy. Moreover, TPO describes the medical contraindication as well as side effects expressed as conditions.

Focusing on treatment personalization, TPO includes concepts related to the patient profile such as diseases, medical conditions including the allergies, the monitored signs, the family history, the current treatments and preferences. Likewise, to enable the dynamic adaptation, our TPO introduces the *TargetGoal* class which is specific to the patient. This goal is fixed by the physician and strongly depends on the patient medical conditions and disease severity stage. It will be used by the *Analysis* process to detect anomalies. Each anomaly is associated to a medical condition in order to be explored by the *Plan* process to extract the interventions that address the detected anomaly. TPO has been conceived to allow the medical experts populating the decision rules through creating relations among TPO instances. Thus, decision rules can be easily updated by experts and reused by the autonomic processes to detect anomalies and generate personalized recommendations. This fruitful model, implemented in OWL [9], can be extended and instantiated for other chronic disease.

To hide the complexity of annotating the medical interventions, we developed a collaborative user-friendly platform that allows medical experts sharing medical knowledge. Our collaborative platform is based on Semantic MediaWiki¹(SMW). SMW unleashes the power of wikis for collaborative knowledge management, and ontologies for providing a common understanding of the domain. It offers mapping mechanisms for formalizing annotations embedded in wiki pages into OWL DL ontology language [10]. We installed around 30 extensions to guarantee an easy interaction with the experts, and to provide visualization services. Our Semantic Web platform also integrates OpenLDAP² to control the access to the patient data. We extended our platform with Fuseki³, a RDF store in order to provide a SPARQL endpoint that allows the external autonomic processes automatically seek the appropriate information. The elaborated SPARQL queries are generic and parameterized. They do not integrate hard-coded conditions related to the patient or to the disease, which makes the autonomic processes reusable.

III. USE CASE: DIABETES MANAGEMENT

In this section, we demonstrate the efficiency of our adaptive system through the ability of the autonomic processes to retrieve the appropriate information from TPO in order to automatically detect the patient medical anomalies, and select personalized recommendations when managing the *hyperglycemia in type 2 diabetes*. The first step consists in using our collaborative platform for annotating the medical interventions and strategies published in the *American Diabetes Association (ADA) and European Association for the Study of Diabetes (EASD)* [8].

A. Use Case Description

We referred to a case study published in the *Clinical Diabetes* journal [11] in order to simulate our system, and compare the recommendations that our system proposes to what the use case claims. The use case describes an elderly patient who is having type 2 diabetes for 11 years. He has

obesity, and his medical history includes hypertension treated with lisinopril, hyperlipidemia treated with pravastatin, right-knee osteoarthritis, a right hip replacement at the age of 61 years, pneumothorax at the age of 35 years, and benign prostatic hypertrophy. He doesn't present any complication from his diabetes, and prefers oral drug than injection. As a first treatment, the physician prescribed for him *metformin* (1000 mg twice daily). According to the use case description [11], the patient HbA1c target goal is fixed to 7%.

B. Simulation of the Adaptation Process

We registered the patient having the aforementioned profile in our platform. We deployed a MAPE loop that simulates the dynamic adaptation based on the patient context changes. The *Monitoring* process periodically measures the patient HbA1c parameter. The frequency of the monitoring is configured according to the patient medical conditions and to the measured parameter. The *Analysis* process retrieves the patient *TargetGoal* from the populated knowledge (in our case it is equal to 7%). If the monitored data is greater than the last measurement and greater than the patient target goal, the *Analysis* process detects an anomaly. This anomaly is associated to the *high blood sugar* medical condition, and is automatically sent to the *Plan* process. In this paper, the *Plan* process operates on the medical knowledge annotated based on TPO and generates the right recommendations based on the disease risk factors, the medical interactions and side effects, the patient profile and preference as the main criteria for the treatment personalization. Figure 2 illustrates an excerpt of the MAPE loop enactment. Our system is able to argue its selection in order to assist the physicians.

```

*****MAPE Loop 1 *****
-----Monitoring-----
The monitored data: 7.7
-----Analysis-----
The Last HbA1c measurement: 7.7; The patient fixed goal is : 7.0
No Problem has been detected
*****End of the Loop 1 *****
*****MAPE Loop 2 *****
-----Monitoring-----
The monitored data: 7.7
-----Analysis-----
The Last HbA1c measurement: 7.7; The patient fixed goal is : 7.0
No Problem has been detected
*****End of the Loop 2 *****
*****MAPE Loop 3 *****
-----Monitoring-----
The monitored data: 8.0
-----Analysis-----
The Last HbA1c measurement: 7.7; The patient fixed goal is : 7.0
=> Anomaly is detected: High blood sugar ...
-----Planning-----
Recommended Interventions
1. Metformin combined with DPP-4inhibitor
2. Metformin combined with GLP-1 receptor agonists
The not recommended interventions are the following:
Metformin-2BInsulin
  • Patient medical condition-SideEffects: Weight_gain
  • Patient medical RiskFactors: Weight_gain; Hypoglycemia
Metformin-2BSulfonylurea
  • Patient medical condition-SideEffects: Weight_gain
  • Patient medical RiskFactors: Weight_gain; Hypoglycemia
Metformin-2BT2D
  • Patient medical condition-SideEffects: Weight_gain
  • Patient medical RiskFactors: Weight_gain; Heart_Failure
-----Execution-----
Notify the physician having the ID:0987Ey
The list of ranked recommendations according to the patient preferences are:
1. Metformin combined with DPP-4inhibitor
2. Metformin combined with GLP-1 receptor agonists
*****End of the Loop 3 *****

```

Figure 2. An excerpt of the autonomic processes enactment

According to the use case description [11], the patient has obesity and hypertension history, interventions that include side

¹ Semantic MediaWiki : <https://www.semantic-mediawiki.org/>

² OpenLDAP: <http://www.openldap.org/>

³Fuseki: https://jena.apache.org/documentation/serving_data/

effects or risk factors such as weight gain and heart failure should be avoided. Moreover, the patient is at an early stage of the hyperglycemia, interventions that may cause hypoglycemia such as the insulin and sulfonylurea should be avoided. The Plan process retains two recommendations, (1) *metformin combined with DPP-4-inhibitor (oral drugs)* and (2) *metformin combined with GLP-1 receptor agonists (oral+injection)*, which are ranked according to the patient preference (oral). The ranked list is sent to the appropriate physician to validate or modify the treatment, and contact his patient.

IV. RELATED WORK

Many research activities investigated the medical knowledge formalization based on Semantic Web for the development of CDSS [12]. However, few studies focus on the dynamic adaptation of the patient treatment based on the context changes, which requires more flexible representation of the medical knowledge. Alexandrou et al. [13] proposed an Adaptive Clinical Pathway Ontology for adapting the patient clinical workflow. The ontology is implemented in OWL and the adaptation process is based on SWRL rules, while the knowledge acquisition is done by experts through Protégé. Similarly, Yao and Kumar [14] proposed to use Protégé to acquire the clinical context ontology and SWRL rules to generate recommendations based on the patient context changes. Their work support the patient treatment generation and prescription checking based on rules. However, these rules are static and hard-coded. In this context, we proposed the TPO schema that overcomes these problems and allows the autonomic processes automatically exploring the annotated knowledge based on flexible SPARQL queries and supporting the integration of external linked data such as DrugBank. Moreover, we developed a collaborative user-friendly interface that interacts with the medical experts to semantically annotate the medical interventions based on TPO.

Lasierra et al. [15] presented an interesting work that relies on autonomic computing to manage patients with chronic disease. The authors used ontology to enable semantic interoperability when monitoring the patient at home using medical sensors. However, their work is interested in the monitoring and analysis processes in order to identify alarms and send warnings to physicians and patients. Both Alexandrou et al. [13] and Lasierra et al. [15] did not consider encoding the characteristics of the medical interventions, which are the basis for the decision support and treatment personalization. Our proposed adaptive system aims at accelerating the decision making and continuously supervising the patient to timely take the right decision. It includes a semantic formalization of the medical interventions based on TPO, which plays the role of a mediator between medical experts and the autonomic processes.

V. CONCLUSION

We proposed an ontology-driven adaptive system for the smart management of the patient treatment based on autonomic computing. The adaptation is based on TPO which is a flexible semantic representation of the medical interventions and their characteristics that underpin the personalization of the patient treatment. To accelerate the decision-making and avoid health

complications, we developed also a set of autonomic processes that rely on TPO to automatically detect medical problems and generate the appropriate recommendations at the right time. As proof of concept, we demonstrated the efficiency of our adaptive system through managing hyperglycemia scenario.

Currently, we are working on an advanced planning algorithm that integrates TPO with existing large scale knowledge sources such as DrugBank. Furthermore, we aim at evaluating the quality of our system through simulating multiple use cases based on experts' collaboration.

ACKNOWLEDGMENT

This research is entirely funded by the National Research Fund (FNR) of Luxembourg under the AFR project.

REFERENCES

- [1] P. A. De Clercq, J. A. Blom, H. H. Korsten and A. Hasman. "Approaches for creating computer-interpretable guidelines that facilitate decision support". *Artificial intelligence in medicine*, vol. 31, 2004, pp 1-27.
- [2] T. Berners-Lee, J. Hendler and O. Lassila. "The semantic web". *Scientific american*, vol. 284, 2001, pp 28-37.
- [3] M. Samwald, A. Jentzsch, C. Bouton, C. S. Kallesøe, E. Willighagen, J. Hajagos, M. S. Marshall, E. Prud'hommeaux, O. Hassanzadeh and E. Pichler. "Linked open drug data for pharmaceutical research and development". *Journal of cheminformatics*, vol. 3, 2011, pp19
- [4] A. Callahan, J. Cruz-Toledo, and M. Dumontier. "Ontology-based querying with Bio2RDF's linked open data". *Journal of biomedical semantics*, 2013, 4(Suppl 1), S1.
- [5] J. O. Kephart and D. M. Chess. "The vision of autonomic computing". *Computer*, vol. 36, 2003, pp 41-50.
- [6] E. Mezghani, M. Da Silveira, C. Pruski, E. Exposito, and K. Drira. "A perspective of adaptation in healthcare". *Studies in health technology and informatics*, In : MIE. 2014, pp 206-210.
- [7] E. Mezghani, E. Exposito, and K. Drira. "A collaborative methodology for tacit knowledge management: Application to scientific research". *Future Generation Computer Systems*, vol. 54, 2016, pp 450-455.
- [8] S. E. Inzucchi, R. M. Bergenstal, J. B. Buse, M. Diamant, E. Ferrannini, M. Nauck, A. L. Peters, A. Tsapas, R. Wender and D. R. Matthews. "Management of hyperglycemia in type 2 diabetes: a patient-centered approach position statement of the American Diabetes Association (ADA) and the European Association for the Study of Diabetes (EASD)". *Diabetes care*, vol.35, 2012, pp1364-1379.
- [9] *Treatment Plan Ontology*. <http://homepages.laas.fr/emezghan/TPO.owl>.
- [10] M. Krötzsch, D. Vrandečić and M. Völkel. "Semantic mediawiki". In : *The Semantic Web-ISWC 2006*. Springer Berlin Heidelberg, p. 935-942.
- [11] S. W. Lahiri. "Management of type 2 diabetes: what is the next step after metformin?" *Clinical Diabetes*, vol. 30, 2012, pp 72-75.
- [12] D. Riaño, F. Real, J. A. López-Vallverdú, F. Campana, S. Ercolani, P. Mecocci, R. Annicchiarico and C. Caltagirone. "An ontology-based personalization of health-care knowledge to support clinical decisions for chronically ill patients". *Journal of biomedical informatics*, vol. 45, 2012, pp 429-446.
- [13] D. A. Alexandrou, I. E. Skitsas and G. N. Mentzas. "A holistic environment for the design and execution of self-adaptive clinical pathways". *Information Technology in Biomedicine, IEEE Transactions on*, vol. 15, 2011, pp 108-118.
- [14] W. Yao and A. Kumar. "CONFlexFlow: Integrating flexible clinical pathways into clinical decision support systems using context and rules". *Decision Support Systems*, vol. 55, 2013, pp 499-515.
- [15] N. Lasierra, A. Alesanco, S. Guillén and J. Garcia. "A three stage ontology-driven solution to provide personalized care to chronic patients at home". *Journal of biomedical informatics*, vol. 46, 2013, pp 516-52

Cross-Model Traceability for Coupled Transformation of Software and Performance Models

Nariman Mani, Dorina C. Petriu, Murray Woodside
Department of Systems and Computer Engineering, Carleton University
Ottawa, Ontario, Canada
{nmani | petriu | cmw}@sce.carleton.ca

Abstract— In Model Driven Engineering, the relationship between a source and target model can be maintained, when the source model undergoes changes, by a coupled transformation, whereby changes applied to the source model are incrementally propagated to the target model. Cross-model traceability links are key to applying the correct changes to the target model. The coupled transformation considered in this paper propagates changes to a Layered Queueing Network (LQN) performance model (originally derived from a UML design model of a SOA system) as an effect of applying design patterns to the SOA model. A special problem arises because of differences in the level of abstraction between UML and LQN (i.e. a performance model element may represent a set of many design model elements). This paper bridges the abstraction gap between models by proposing traceability links that use new collection types (not defined in the source metamodel) to represent complex source model elements, which are then mapped to simple target model elements.

Keywords- *Software performance model, service oriented systems, SOA pattern, traceability links, change propagation, LQN*

I. INTRODUCTION

In Model Driven Engineering (MDE), the performance of a Service Oriented Architecture (SOA) design can be evaluated in early lifecycle phases using a quantitative performance model (hereafter called the *PModel*) generated by a model transformation from the software design model (hereafter called the *SModel*) extended with performance annotations. An example of such a technique is the Performance from Unified Model Analysis (PUMA) [1][8]. Also, system designers often apply SOA patterns [2] to system designs as generic solutions for architectural, design and implementation problems, and study their performance impact with the help of the corresponding PModel. A pattern could have a significant performance cost due to the overheads it may introduce, in which case the performance cost can be balanced against the benefits of the pattern, and alternative pattern configurations can be compared. Traditionally, a new performance model needs to be generated by reusing the PUMA technique to evaluate the impact of the design pattern changes on the design model. However, this has drawbacks: **1)** it masks the causal connections between the design changes and the performance impact which can provide significant insight to the engineer to make design choices; **2)** it is a substantial waste of execution cost, which could be significant if the cycle of choosing a pattern, applying and evaluating it is repeated

many times during the development process of large systems. Cross-model traceability can help by maintaining consistent relationships between the source and target model elements from the moment the target model is generated; when changes occur in the source model, only the affected target model elements are identified and changed. Compared to generating a new performance model by techniques such as PUMA every time a pattern is applied, the cross-model traceability links show the causal connections between the SModel changes and the resulting PModel changes with a reduced effort. It also enables incremental studies of numerous design alternatives when applying a large number of SOA design patterns.

Cross-model traceability links are straightforward when the cross-model relationships between elements are one-to-one, or one-to-many. However, when generating a PModel from a SModel, it often happens that one PModel element is created from a set of SModel elements, due to the fact that the level of abstraction of PModel is higher. An example is a collection of SOA activities which make up a single service operation in the PModel. The requirements for these specific collection elements are defined in Section V and VI.

In [3] we proposed a coupled refactoring technique which incrementally propagates the SModel changes (due to the application of a SOA design pattern) to the PModel. It uses entity-to-entity traceability links without considering the abstraction gap and makes it the responsibility of the designer to identify a collection of SModel entities which all trace to the same PModel entity. This process is error-prone and requires deep designer understanding of the process.

The difference in this paper is that we address the challenge of bridging the abstraction gap between the two models by defining new types for complex source model concepts (not corresponding to any meta-class in the source metamodel) and mapping them to the target model concepts. Moreover, in this paper we propose an improved (and in fact simplified) version of the coupled transformation process in [3] based on these extended types, as explained in sections VI and VII. This makes the coupled transformation more accurate and easier to automate. Furthermore, the extended types can also be used to trace the performance results obtained by solving the PModel back to the collections of SModel elements corresponding to the PModel elements.

In this paper, the SModel uses UML extended with the SoaML profile [4, 5] for SOA concepts and the MARTE profile

(Modeling and Analysis of Real-Time and Embedded systems) [6] for performance-related information. The PModel is expressed in the Layered Queueing Network (LQN [7]) formalism. The initial PModel is created from the annotated UML using the PUMA tools [1, 8]. All of these models are briefly described in Section IV.

II. RELATED WORK

Traceability is frequently employed in approaches to software model transformation. In [9], the authors present a method which attaches traceability generation codes to pre-existing ATL programs [10]. The method produces a loosely coupled traceability, meaning it can be used for any kind of one-to-one traceability. In [11] a method is presented for generating annotated models with traceability information, by merging the models with the trace models. The generated trace-links are embedded in the target model, in elements they refer to, or are stored externally in a separate model.

Managing the complexity of traceability information in MDE is discussed in [12]: a) how to identify different kinds of trace-links that may appear in MDE; and b) propose a rigorous approach for defining semantically rich trace-links between models. In [13] the authors propose a traceability framework, implemented in the model-oriented language Kermeta, to facilitate modeling transformations. Using a trace metamodel, the framework allows for tracing the transformation chain within Kermeta. Model transformation trace-links are defined in the metamodel as a set of source nodes and target nodes.

None of the above works addresses traceability between models at different levels of abstraction. On the other hand, reverse engineering transformations, which do raise the abstraction level, do not emphasize traceability, perhaps because in reverse engineering there is less interest in retaining the connection with the original model. In reverse engineering of design models from code [14] a single design element may be represented by many scattered features of the code, with structured relationships which must be captured in the traceability link. The taxonomy from [14], for example, does not mention traceability links. However, coupled transformations of software and performance models, such as our proposed techniques in [3], require constructing and maintaining these links. Therefore, the technique proposed in this paper that addresses the abstraction gap between software and performance models, does improve our previous approach from [3].

III. OVERVIEW OF THE IMPROVED APPROACH

Figure 1 shows an overview of the coupled transformation technique [3], enhanced with extended types for traceability links introduced in this paper for propagating changes due to design patterns. The inputs to the process include the initial SOA SModel (top left), and a library of pattern definitions (bottom left). The enhanced traceability links are used in stage C (shown in grey) for translation of the SModel refactoring transformation rules into PModel refactoring transformation rules. The designer steps (supported by tools developed by the

authors) are shown on the left side and the automated steps on the right side.

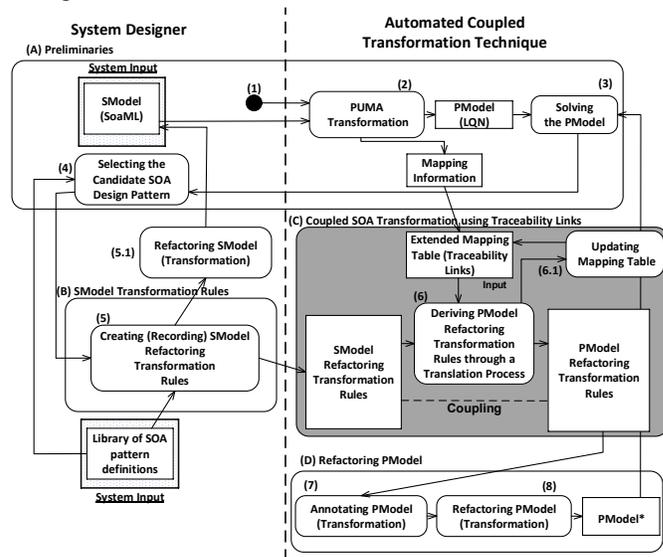


Figure 1: Overview of Improved Approach using Extended Traceability Links for Coupled Transformations

The extended approach has four stages:

A. Preliminaries: This stage gets the SModel as an input and creates the base PModel using PUMA [1]. The mapping information between the SModel and the constructed PModel is created during this initial transformation process (Step (2)). The mapping information is used by the technique in this paper for creating the cross-model traceability links with extended types in a mapping table which will be used in stage C. Pattern application begins at step (4), where the designer selects a candidate pattern for its own reasons (e.g. maintainability).

B. Model Transformation Rules: The selected pattern is specified using Role-Based Modeling RBML [15], a graphical pattern specification approach which uses model roles to identify the participating elements. The designer indicates where the pattern is applied by binding pattern roles to elements in the SModel and then records SModel transformation rules that will satisfy the solution specification (step (5)).

C. Deriving the PModel Transformation Rules: Using the traceability links with extended type extracted from the mapping information generated in Stage (A) and the SModel refactoring transformation rules from Stage (B), the PModel refactoring transformation rules are derived automatically in the coupled transformation process (Step 6). The dashed line between the “SModel Refactoring Transformation Rules” and “PModel Refactoring Transformation Rules” represents the coupling between them. If there are updates to the table of traceability links (mapping table) due to add/deletion of the elements, this is being done as part of Step 6.1.

D. Refactoring PModel: The PModel refactoring transformation rules are executed by a transformation engine to refactor the PModel into the final PModel* (Steps 7 and 8). Although the steps in this stage are explained briefly in Section VII.B, the details are discussed in [3] and are not

within the scope of this paper as they are not impacted by the proposed extended types in this paper.

IV. MODELS

A. SOA Models

From the range of views in SoaML [4, 5] used to model SOA systems, we use the Business Processes Model (BPM) for behavior and the Service Architecture Model (SEAM) for structure and contracts, together with a UML deployment diagram. The SEAM is specified as a UML collaboration diagram with service participants and contracts (with SoaML stereotypes *«Participant»* and *«ServiceContract»* respectively). Each participant plays a role of Provider or Consumer with respect to a contract. Participants correspond to pools, participants and swimlanes in the BPM. The BPM is specified as a UML Activity Diagram (AD) (see Figure 2). Service invocations are modeled as operation calls, using three types of UML actions: a *CallOperationAction* sends a service request and waits for the reply via its input/output pins; an *AcceptCallAction*, an accept event action, waits for the request arrival; and a *ReplyAction* returns the reply values to the caller. The called operation name appears in ‘()’ as “(class-name::operation-name)”. We assume that all BPM edges between ActivityPartitions represent calling interactions, connecting these three types of Actions.

MARTE performance annotations are given in shaded notes. BPM describes the behavior as a sequence of steps *«PaStep»* with a workload attached to the first step stereotyped as *«GaWorkloadEvent»*. *«PaStep»* has attributes *hostDemand* (required CPU time), *rep* (mean repetitions) and *prob* (probability of optional step). *«GaWorkloadEvent»* defines a population of *Nusers* users, each with a thinking time *ThinkTime* defined by MARTE variables. Concurrent runtime instances *«PaRunTInstance»* are identified with swimlane roles. UML Deployment Diagram (DP) is also defined, as in Processing nodes are stereotyped *«GaExecHost»* and

communication network nodes are stereotyped *«GaCommHost»*, with attributes for processing capacity, message latency and communication overheads.

B. Performance Model

PModels are expressed in an extended queuing notation called Layered Queuing Networks (LQNs) [8], selected because of its close coupling to the high-level software architecture. An LQN estimates waiting for service due to contention for host processors and software servers, and provides response time and capacity measures. Figure 3 shows the LQN model for the example. For each service there is a task, shown as a bold rectangle, and for each of its operations (contracts) there is an entry, shown as an attached rectangle. The task has a parameter for its multiplicity or thread pool size (e.g. {‘1’}). Each entry has a parameter for its host CPU demand, equal to the total *hostDemand* of the set of *«PaSteps»* for the same operation in the SModel.

Calls from one entry to another are indicated by arrows between entries (a solid arrowhead indicates a synchronous call for which the reply is implicit, while an open arrowhead indicates an asynchronous call). The arrow is annotated by the number of calls per invocation of the sender. For deployment, an LQN host node is indicated by a round node associated to each task. While Figure 3 shows entries with host demands and calls, there is an optional level of detail which is not shown here, which defines an activity subgraph for each entry with predecessors, successors, forks and joins, similar to a UML activity diagram. The host demands and calls are then defined for each activity.

V. THE ABSTRACTION GAP BETWEEN SModel AND PModel

Each type of traceability link produced by the SModel-to-PModel transformation describes the mapping relationship between a SModel element type (i.e., a meta-class of the UML metamodel) and a PModel element type (i.e., a meta-class of the LQN metamodel). Establishing the traceability links

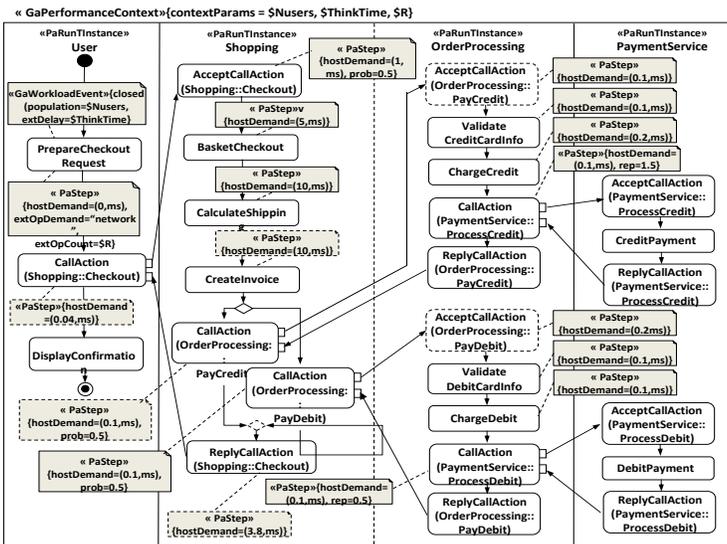


Figure 2: Checkout Business Process Model for the Online Shop

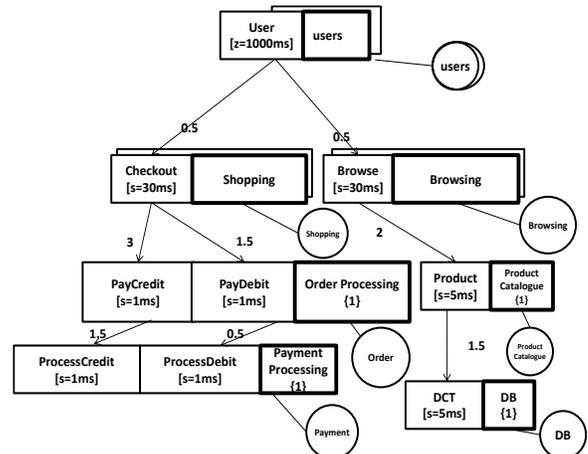


Figure 3: LQN (PModel) corresponding to SOA design (SEAM, BPM and Deployment Diagram)

between elements of SModel and PModel which are in a one-to-one relationship is straightforward. A partial list of one-to-one mappings is shown in the following table:

SModel (SoaML) Type	PModel(LQN) Type
SEAM Participant	Task
BPM Swimlane (PaRunTInstance)	Task
BPM Action	Activity
BPM Control Flow	Sequence
BPM Async Call	Async Call
DP Processing Node (ExecHost)	Host
DP Artifact	Task

However, SModel to PModel relationships are not always one-to-one. We have identified cases where a group of interconnected SModel elements (called subgraph) is mapped to one or more PModel elements (i.e. many-to-one or many-to-many relationship). In most cases, there are more SModel elements mapped to fewer PModel elements, indicating that the latter has a higher level of abstraction. These mappings are described below.

A. LQN Entry

An LQN entry of a task represents the entire operation carried out by the task in response to a call. Thus, the entire subgraph of SModel activity diagram actions, control flows, hyper edges and attributes invoked by an AcceptCallAction is mapped to an LQN entry. An example of this type of mapping is shown in Figure 4.

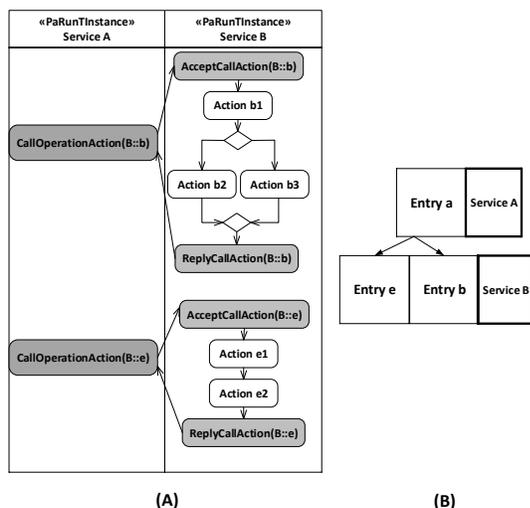


Figure 4: (A) Partial SModel AD Subgraph and (B) Corresponding Partial LQN PModel with mapped Entry

The partial SModel AD and the corresponding partial LQN PModel are shown in Figure 4.A and Figure 4.B, respectively. There are two swimlanes representing “Service A” and “Service B”, where “Service B” contains two subgraphs, each accepting a call from a CallOperation-Action in “Service A”. The subgraph between *AcceptCallAction(B::b)* and *ReplyAction(B::b)* is mapped to “Entry a”, and the one between *AcceptCall-Operation(B::e)* and *ReplyAction(B::e)* to “Entry e” in the partial LQN model shown in Figure 4.B. This is a case of many-to-one mapping.

B. LQN Synchronous Call

An LQN synchronous call corresponds to two messages in the SModel, the call and the corresponding reply. This is also a case of many-to-one mapping. Two examples of this type of mapping are shown in Figure 5. The synchronous call from *CallOperationAction(B::b)* in “Service A” to *AcceptCallAction(B::b)* in “Service B” and the reply from *ReplyCallAction(B::b)* to the caller action in Figure 5.A are mapped to a single LQN call from “Entry a” to “Entry b” in Figure 5.B. Figure 5 shows also an example of nested synchronous calls, where a call to “Service C” is made before the reply from “Service B” to the initial caller, “Service A”.

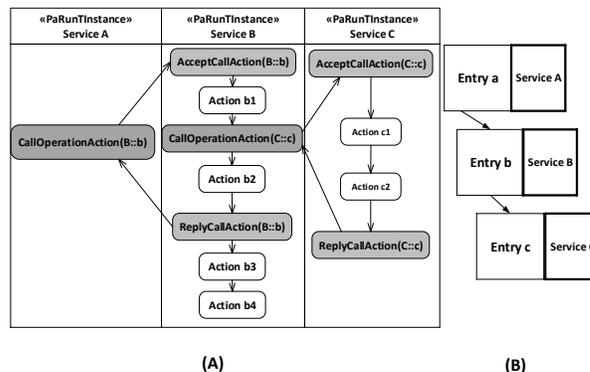


Figure 5: (A) Partial SModel AD with synchronous calls and (B) Corresponding LQN PModel with synchronous calls

C. LQN Asynchronous Call

An LQN asynchronous call represents a SModel call without reply, which also might be part of a forwarding call (to be dealt with next). While this correspondence is one-to-one, it is mentioned here because it only occurs only in a certain context in the SModel.

D. LQN Forwarding Call

An LQN forwarding call represents delegation of responsibility for an operation. It is a chain of calls in the PModel that corresponds to a chain of messages in the SModel. There is an initial synchronous call from a *CallOperationAction* which eventually receives its corresponding reply from a different swimlane than the one it called, and one or more asynchronous calls that forward the caller request to another swimlane; the final one in the chain replies to the initial caller.

This collection of SModel messages is mapped to the following collection of PModel elements: a LQN synchronous call and one or more forwarding calls that forward the request to a final entry, which implicitly provides the reply to the initial caller. This is a case of many-to-many mapping between the SModel and PModel elements. Figure 6 shows an example. *CallOperationAction(B::b)* from “Service A” in Figure 6 .A initiates a call to “Service B”, which forwards it to “Service C”, which replies to the initial caller. Figure 6.B shows corresponding (mapped) partial LQN PModel with one synchronous call and one forwarding call (i.e. dashed arrow line).

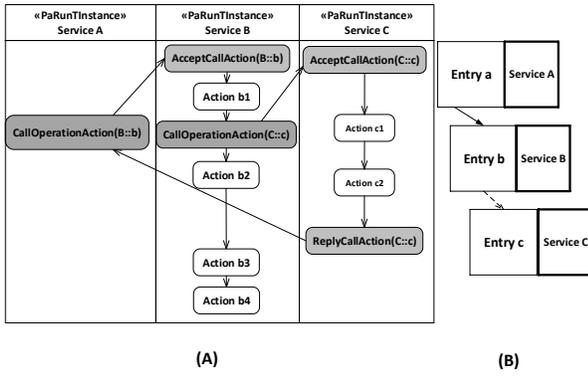


Figure 6: (A) Partial SModel AD forwarding scenario and (B) Corresponding Partial LQN PModel with synchronous & forwarding calls

VI. TRACEABILITY LINKS AND MAPPING TABLE

We assume that the transformation that derives the initial PModel from the SModel (e.g. PUMA) also generates the basic element-to-element mapping between SModel and PModel elements (i.e. in form of a mapping table), as described in [16]. In this paper, the initial mapping table provided by PUMA is extended with the additional higher-level types of traceability link, as described in this section. Mapping table is discussed in Section VI.A and the traceability links metamodel is described in Section VI.B.

A. Mapping Table

The mapping table is a collection of the traceability links of the form:

$Traceability\ Link = (link\ name, SME, PME)$

where SME stands for an SModel Element or attribute, and PME for the corresponding PModel Element or attribute. For example, the traceability link named BTL1 between the BPM swimlane “Service B” in Figure 5.A and the LQN Task “Service B” in Figure 5.B has the form:

$Traceability\ Link = (BTL1, BPM::Swimlane:ServiceB, LQN::Task:ServiceB)$

B. Traceability Links Metamodel

To bridge the abstraction gap, in this paper we propose that traceability links in the mapping table use a metamodel that includes the following:

- UML types describing the SModel elements,
- LQN metamodel types describing the PModel elements,
- the following four additional types:
 1. *EntrySME*: a collection of Elements in the SModel containing the elements of the Activity Subgraph invoked by a Call,
 2. *SyncCallSME*: a pair of Call and Reply Actions and their corresponding AcceptCallActions in SModel, that make up a synchronous call
 3. *ForwardingSME*: a collection of CallActions and one ReplyAction in SModel, with their corresponding AcceptCallActions, forming a forwarding pattern as described in Section V.
 4. *LQNFwdCall* or *ForwardingPME*: a corresponding collection of LQN calls in PModel with one synchronous call and one or more forwarding calls.

Therefore, the SME column of the mapping table has UML types plus four additional types for SModel and PME column has one additional type for PModel. The traceability links based on these extended types are called “traceability links with extended types” and are defined in the following sub sections:

1) Traceability Link for an LQN Entry

An entry in the PModel corresponds to a portion of the behaviour specified in an activity diagram, defining the response by a PaRunTInstance (defined by a swimlane) to a call. The subgraph starts with an *AcceptCallAction* following a call from a *CallOperationAction* in another swimlane, and ends where it provides a reply with a *ReplyAction*, or ends, or executes a *CallOperationAction* to another swimlane.

The EntrySME can be discovered automatically by an analysis of the flow of Actions, based on this definition. In presenting it here, it is shown as a list of the elements in the subgraph within ‘{}’ brackets, separated by commas. In this list, the action names are shown first, then the activity control flows each defined as a couple (source, destination) and finally the hyper edges (Decision, Merge, Fork, Join). Each hyper edge is defined as:

<i>Type</i> (predecessor list, successor list) if <i>Type</i> is Decision or Fork , there is only one predecessor; if <i>Type</i> is Merge or Join , there is only one successor.

To distinguish the hyperedges in the text, their types are shown in bold.

The EntrySME is mapped to a LQN Entry and each element in the defining list is mapped to a corresponding element of the LQN activity subgraph inside the LQN entry. An example of this type of traceability link, referencing elements in the scenario in Figure 4, is given below:

$Link = (BTL2, BPM::EntrySME: \{AcceptCallAction(B::b), Action(B::b1), Action(B::b2), Action(B::b3), ReplyAction(B::b), (AcceptCallAction(B::b), Action(B::b1)), \mathbf{Decision} (Action (B::b1), Action(B::b2), Action(B::b3)), \mathbf{Merge} (Action (B::b2), Action(B::b3), ReplyCallAction(B::b))\}, LQN::LQN\ Entry: \{(Entry\ b)\})$

2) Traceability Link for an LQN Synchronous call

A synchronous call (i.e., call-reply) in the PModel corresponds to a pair of messages (that is, of ActivityEdges that cross the boundary between two ActivityPartitions), called here a Synchronous Call. The first message is from a *CallOperationAction* in the first BPM swimlane to an *AcceptCallOperation* in the second swimlane; the second message is from a *ReplyAction* in the second swimlane to the initiating *CallOperationAction*. A SyncCallSME is defined as a two-element list as follows:

$\{(CallOperationAction, AcceptCallOperation), (ReplyAction, CallOperationAction)\}$
--

which is mapped to the corresponding LQN Synchronous Call. An example of this type of traceability link based on the scenario in Figure 5 is given below:

$Link = (BCTL3, BPM::SyncCallSME: \{(CallOperationAction(B::b), AcceptCallAction(B::b), (ReplyAction(B::b), CallOperationAction(B::b))\}, LQN::LQNSyncCall: \{(Entry\ a, Entry\ b)\})$
--

Table 1: Examples of corresponding SModel and PModel element types in the traceability links of the Mapping Table

Sub-table (A) Types for Structural Elements		
	SME	PME
C1	Participant (in SEAM)	LQN Task
C2	Host Node (in Deployment)	LQN Host
C3	ActivityPartition/Swimlane (in BPM) stereotyped «PaRunTInstance»	LQN Task
C4	EntrySME (collection of elements forming an activity subgraph)	LQN Entry
C5	BPM Action	LQN Activity
Sub-table (B) Calls		
C6	SyncCallSME (collection of calls)	LQN Synchronous Call/LQNSyncCall
C7	ForwardingSME (collection of calls)	LQN Forwarding Call/LQNFwdCall (collection of calls)
C8	Asynchronous Call	LQN Asynchronous Call/LQNAsyncCall
Sub-table (C) Attributes		
C9	MARTE WorkloadEvent.extDelay	Think Time of a workload
C10	MARTE ExecHost.resMult	Processor Multiplicity
C11	MARTE PaRunTInstance.poolsize	Task Multiplicity

Table 2: Mapping Table with examples of traceability links for Shopping and Browsing SModel and PModel

Sub-table (A) Structural Elements		
Link	SME	PME
DTL3	Deployment Node Order Host	LQN Host Order
DTL2	Deployment Artifact Browsing	LQN Task Browsing
STL3	SEAM Participant User	LQN Task User
STL2	SEAM Participant Browsing	LQN Task Browsing
BTL1	EntrySME: {(AcceptCall (Shopping::Checkout), BasketCheckout, CalculateShipping, CreateInvoice, CallOperation(OrderProcessing::PayCredit), CallOperation(OrderProcessing::PayDebit), ReplyCallAction(Shopping::Checkout), (AcceptCall (Shopping::Checkout), BasketCheckout), (BasketCheckout, CalculateShipping), (CalculateShipping, CreateInvoice), Decision (CreateInvoice, CallOperation(OrderProcessing::PayCredit) , CallOperation(OrderProcessing::PayDebit)), Merge (CallOperation(OrderProcessing::PayCredit) , CallOperation(OrderProcessing::PayDebit), ReplyCallAction(Shopping::Checkout))}	LQNEntry: {(Checkout)}
Sub-table (B) Calls		
BCTL1	SyncCallSME: {(CallOperationAction(Shopping::Checkout), (AcceptCallAction(Shopping::Checkout), ReplyAction(Shopping::Checkout), CallOperationAction(Shopping::Checkout))}	LQNSyncCall: {(User,Checkout)}
Sub-table (C) Attributes		
BATL1	MARTE Attribute <i>hostDemand</i> for the AcceptCallAction(Shopping::Checkout)	Host Demand attribute of LQN AcceptCallAction in Entry Checkout

3) Traceability Link for an LQN Forwarding Call

A Forwarding Call identifies a call pattern which includes synchronous and asynchronous calls. It begins with a synchronous call from a *CallOperationAction* in one BPM swimlane to an *AcceptCallOperation* in a second swimlane. However instead of a reply, this operation ends with a call that forwards the request to a third swimlane. It may be forwarded any number of times, until a reply is sent back to the originating swimlane. The fact that the reply is coming from a swimlane which is different than the swimlane of the receipt of initial call shows that the request has been forwarded to other swimlanes for processing. A ForwardingSME is defined as follow: it begins with *SyncCallSME* (defined above for the

synchronous calls) which is followed by a comma-separated list of forwarding calls, as:

$$\text{ForwardingSME:} \{ \text{SyncCallSME}, (\text{fwdCall1}), (\text{fwdCall2}), \dots \}$$

A ForwardingSME is mapped to a collection of PModel elements containing an LQN synchronous call and one or more LQN forwarding calls. The last forwarding call implicitly generates the reply to the originating entry. This requires another extended type for the LQN column of the mapping table, called *LQNFwdCall*. A *LQNFwdCall* begins with *LQNSyncCall* and it is defined similarly by a list:

$$\text{LQNFwdCall:} \{ \text{LQNSyncCall}, (\text{fwdCall1}), (\text{fwdCall2}), \dots \}$$

An example of this type of traceability link based on the scenario in is:

$$\text{Link} = (\text{BCTL4}, \text{BPM::ForwardingSME:} \{ (\text{CallOperationAction}(\text{B::b}), \text{AcceptCallAction}(\text{B::b}), (\text{ReplyAction}(\text{C::c}), \text{CallOperationAction}(\text{B::b})), (\text{CallOperationAction}(\text{C::c}), \text{AcceptCallAction}(\text{C::c})) \}, \text{LQN::LQNFwdCall:} \{ (\text{Entry a}, \text{Entry b}), (\text{Entry b}, \text{Entry c}) \})$$

Table 1 shows examples of corresponding SModel and PModel element types in the traceability links of the mapping table. Table 1 is organized into three groups for Structural Elements, Calls, and Attributes. Table 2 gives some examples of traceability links established between the SModel (BPM is shown in Figure 2) and PModel (Figure 3) for the shopping and browsing SOA.

VII. COUPLED TRANSFORMATION USING EXTENDED TRACEABILITY LINKS

The coupled transformation technique in [3] includes a formal recording of the refactoring of the SModel, and automatic derivation of the refactoring transformation of the PModel as well as its automatic application to the PModel. This section describes how these steps must be modified (enhanced) to accommodate the extended types of cross-model traceability links with extended types proposed in this paper. An overview of the modified process is also shown in Figure 1 (Stages B and C).

A. Coupled PModel Refactoring Rules

The coupled transformation in [3] begins with recording the SModel refactoring that arises from the pattern application. A tool was implemented in [3] to assist the system designer with this process. With the extended link types introduced in this paper, the system designer can now create rules at various levels of abstraction. Therefore in this paper, the tool in [3] is enhanced to support extended types. The designer creates the rules using the provided tool by selecting SMEs from a table created from the UML specification, and can choose the level of abstraction by choosing from the extended element types. For example, when refactoring behavior, rules can be applied either to an entire EntrySME or to its individual activities, as the designer wishes. The automated translation of the SModel refactoring rules into PModel refactoring rules is based on the cross-model traceability links in the mapping table described in Section VI, using both the Type Correspondences table (e.g. Table 1) and the Mapping Table (e.g. Table 2). With the extended model and link types the process of generating the

PModel refactoring rules is made more accurate, simpler and more uniform. Therefore the automated process in [3] is modified to use the extended types as follow. Figure 7 shows a partial screenshot of the tool that takes care of the enhanced automated translation using the traceability links. Each SModel refactoring rule has an operation name and some arguments, which are processed as follows:

1. The operation name generates one or more PModel operations. The action part of the name (add/ delete/ modify) is retained, and the operand-type part (e.g. Participant) is mapped according to the Type Correspondences table, similar to one shown in Table 1 For example the SModel operation addParticipant is translated to addTask, and deleteEntrySME to deleteEntry.
2. The arguments of the PModel operation (e.g. the element or elements to be added, deleted, or modified) are translated from the arguments of the SModel operation using the Mapping Table (e.g. Table 2). In an “add” operation the name of the new PModel element is taken as the name of the corresponding SModel element.

For example, a SModel “addParticipant” operation is mapped to “addTask” for the PModel, and the “addParticipant” argument becomes the new task name. Modifications to calls require special consideration in the translation. The SModel “modifyActionCall” operation changes a service invocation from a *CallOperationAction* to an *AcceptCallAction*. As this might apply to more than one call to the same *AcceptCallAction*, the mapping table is searched (using a MappingTableSearchByKey command) to identify all the PModel activities making the call. Then the operation is mapped to one or more “modifyActivity” operations in the PModel domain, to change all the calls. Some of the PModel transformation rules derived for the Façade pattern are presented in Figure 7.

B. PModel Refactoring

The extended types introduced in this paper do not impact the

process of applying the PModel refactoring rules (stage D in Figure 1) discussed as part of the coupled transformation technique in [3]. Briefly, first the PModel is annotated with transformation directives indicating the changes, then the changes are applied by a transformation engine implemented using QVT Operational (Query, View, and Transformation, a OMG standard model transformation language) which processes the directives. The details are provided in [3].

VIII. CASE STUDY

To illustrate the application of the coupled transformation and the role of the extended types and traceability links in the mapping, two SOA patterns will be applied to the Shopping and Browsing SOA given in Figure 2 and Figure 3.

Suppose a designer must re-design a Shopping and Browsing SOA to support three different user types (mobile phone, desktop, kiosk) through a multi-channel endpoint. First, the designer applies the pattern “Concurrent Contracts”[2], which addresses the following problem: *A service’s contract may not be able to support all potential types of clients, because of access and interface differences.* The pattern also suggests the following solution [2]: *To accommodate different types of clients, separate service contracts (“channels”) can be created for the one underlying service implementation.* Using the Concurrent Contracts pattern, separate shopping and browsing operations are provided for each group of users. Separate sets of actions (in form of activity subgraphs, i.e. EntrySMEs) are created in the Shopping swimlane (see Figure 3) and also the Browsing swimlane (not shown in Figure 3). Using the corresponding types table (using row C4 in Table 1), traceability links are created for these EntrySMEs mapping the activity subgraphs to new PModel LQN entries (LQNEntry) of the Shopping and Browsing tasks. Although this allows each contract to be extended and managed individually, it introduces duplication in the functional design. Furthermore, when a service is subject to change due to contract changes, the core service logic needs to be extended

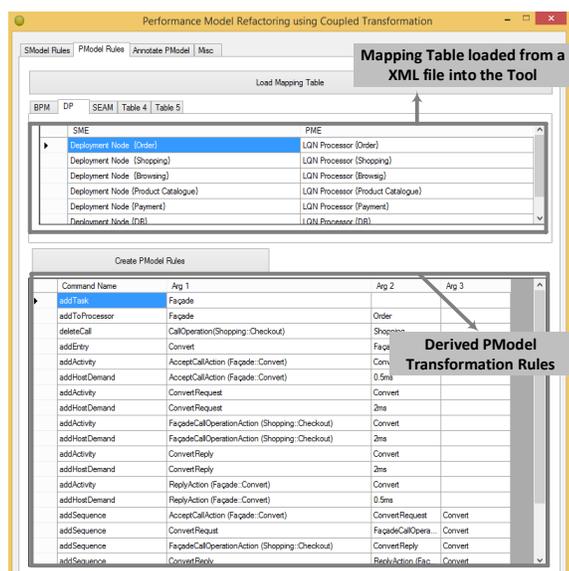


Figure 7: Tool for automatic derivation of PModel refactoring rules

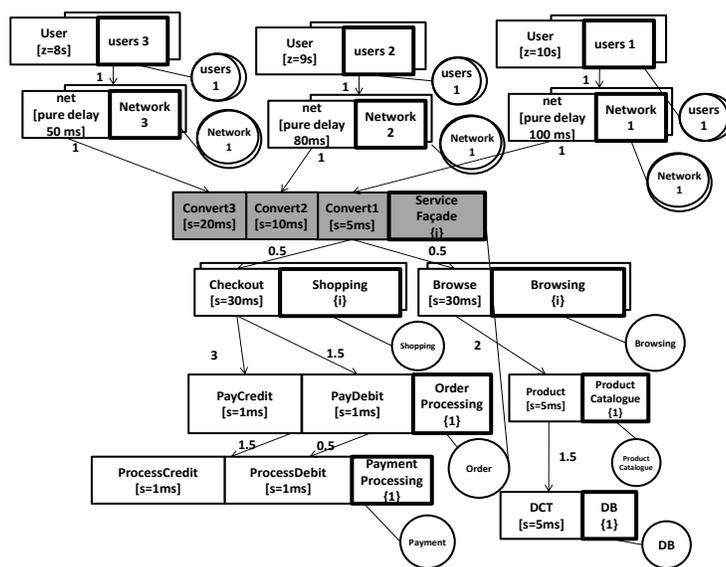


Figure 8: Shopping and Browsing LQN PModel with the Façade Design Pattern

and augmented to accommodate the change. This leads the designer to consider another SOA design pattern called “Service Façade”, which addresses the following problem according to [2]: *The tight coupling of the core service logic to its contracts can obstruct its evolution and negatively impact service consumers.* The solution suggested in [2] is: *Façade logic is inserted into the service architecture to establish a layer of abstraction that can adapt to future changes to the service contract.*

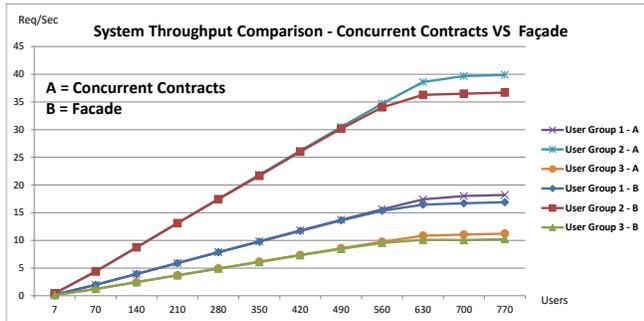


Figure 9: System Throughput (Requests/Sec) for Concurrent Contracts and Façade

To apply the Façade design pattern, a new Façade swimlane is created in the BPM with one new EntrySME activity subgraph for each service contract. The new swimlane is mapped to a new LQN task ServiceFaçade in the PModel (using row C3 in Table 1), and the new EntrySMEs are mapped its entries (LQNEntry) (using row C4 in Table 1). The BPM synchronous calls from the users to the Shopping swimlane are redirected to pass through the façade EntrySMEs. The synchronous calls are mapped to LQN synchronous calls through the traceability links with SyncCallsSME types (C6 in Table 1). The LQN PModel after application of the Façade Design pattern is shown in Figure 8 (PModel before application is shown in Figure 3). The refactored PModel is solved by LQN Solver tool [8] for performance analysis (i.e. throughput, response time, utilization. etc). Figure 9 compares the system throughput (request/sec) given by the LQN model solver for the two cases (Concurrent Contracts and Façade) when the total number of users varies for three types of users. Figure 9 shows that application of Façade design pattern is consistently making the system throughput worse compared to Concurrent Contracts, due to the additional overhead. This shows that Façade has a performance cost to balance against its architectural benefits.

IX. CONCLUSION

Establishing cross-model traceability links is challenging when there is an abstraction gap between the source and target models. In case of the SOA SModel and the corresponding PModel created by the PUMA [1] transformation chain, the abstraction difference involves collections of elements in both models. In this paper, four relationships were identified which involve many-to-one or many-to-many mappings which reveal the gap in the level of abstraction between SModel and PModel. To bridge the abstraction gap, these four additional types for subgraphs of SModel and PModel elements are defined and used in establishing the extended traceability links. These additional types do not correspond to any meta-

class of the source or target metamodel. The syntactic correctness of the corresponding artifacts is verified by the model transformation that generates the target model from the source model, and also identifies the model elements contained in every artifact. We also modified the coupled transformation technique in [3] to use the traceability links with extended types which keeps consistent the SModel and PModel after the application of a SOA pattern. Examples of use are described with coupled refactoring transformations that represent the application of two SOA patterns: “Concurrent Contracts” and “Service Façade”.

ACKNOWLEDGEMENT

This work was supported by the Natural Sciences and Engineering Research Council of Canada (NSERC) through its Discovery Grant program.

REFERENCES

- [1] M. Woodside, D. Petriu, J. Merseguer, D. Petriu, and M. Alhaj, "Transformation challenges: from software models to performance models," *Software & Systems Modeling*, vol. 13, pp. 1529-1552, 2014.
- [2] T. Erl, *SOA Design Patterns* Boston, MA: Prentice Hall PTR, 2009.
- [3] N. Mani, D. Petriu, and M. Woodside, "Exploring SOA Pattern Performance using Coupled Transformations and Performance Models," *the 27th International Conference on Software Engineering and Knowledge Engineering (SEKE 2015)*, Pittsburgh, PA, USA, 2015, pp. 552-557.
- [4] Object Management Group, "Unified Modeling Language (UML)," Version V2.4.1, formal/2011-08-05
- [5] Object Management Group, "Service oriented architecture Modeling Language (SoaML)" Version 1.0.1, formal/2012-05-10
- [6] Object Management Group, "A UML Profile for MARTE (Modeling and Analysis of Real-Time and Embedded systems)," Version 1.1, formal/2011-06-02.
- [7] G. Franks, T. Al-Omari, M. Woodside, O. Das, and S. Derisavi, "Enhanced Modeling and Solution of Layered Queueing Networks," *IEEE Trans. on Software Eng.*, vol. 35, pp. 148-161, 2009.
- [8] M. Woodside, D. C. Petriu, D. B. Petriu, H. Shen, T. Israr, and J. Merseguer, "Performance by Unified Model Analysis (PUMA)," *WOSP '05 Proc. of the 5th international workshop on Software and performance*, Palma de Mallorca, Illes Balears, Spain, 2005, pp. 1 - 12
- [9] F. Jouault, "Loosely Coupled Traceability for ATL. In: Traceability Workshop," *Traceability Workshop at European Conference on Model Driven Architecture (ECMDA-TW)*, Nürnberg, Germany, 2005, pp. 29–37.
- [10] F. Jouault and I. Kurtev, "Transforming Models with ATL," *MoDELS'05 Proc. of the 2005 international conference on Satellite Events at the MoDELS*, Montego Bay, Jamaica, 2005, pp. 128-138.
- [11] D. S. Kolovos, R. F. Paige, and F. A. C. Polack, "On-Demand Merging of Traceability Links with Models," *3rd ECMDA Traceability Workshop*, 2006.
- [12] R. F. Paige., N. Drivalos., D. S. Kolovos., K. J. Fernandes., C. Power., G. K. Olsen., et al., "Rigorous identification and encoding of trace-links in model-driven engineering," *Software and Systems Modeling (SoSyM)*, vol. 10, pp. 469-487, 2011.
- [13] J.-R. e. Falleri, M. Huchard, and C. e. Nebut, "Towards a Traceability Framework for Model Transformations in Kermeta," *Traceability Workshop at European Conference on Model Driven Architecture (ECMDA-TW)*, 2006, pp. 31-40.
- [14] E. J. Chikofsky and J. H. Cross, "Reverse engineering and design recovery: a taxonomy," *Software* vol. 7, pp. 13 - 17, 1990.
- [15] R. B. France, D.-K. Kim, S. Ghosh, and E. Song, "A UML-Based Pattern Specification Technique," *IEEE Trans. Software Eng.*, vol. 30, pp. 193-206, 2004.
- [16] M. Alhaj and D. Petriu, "Traceability Links in Model Transformations between Software and Performance Models," in *SDL 2013: Model-Driven Dependability Engineering*. vol. 7916, F. Khendek, M. Toeroe, A. Gherbi, and R. Reed, Eds., Springer, 2013, pp. 203-221.

Exploring the Dimensions of Electronic Government Service Quality

Dr. Taiseera Hazeem Al Balushi

Assistant Professor, Department of Information Systems,
College of Economics and Political Science
Sultan Qaboos University
Muscat, Oman
taisira@squ.edu.om

Dr. Saqib Ali

Associate Professor, Department of Information Systems,
College of Economics and Political Science
Sultan Qaboos University
Muscat, Oman
saqib@squ.edu.om

Abstract—Information and communication technology is progressively evolving around the world. Organizations and providers of online services, have to keep up with technological advancements when providing services and meet the redoubling expectations of its customers. At a global level there is wider adoption of e-government, a term coined in the recent past to describe the methodology that implements technology efficiently with quality of service for the general public's administration in a government setting. This research proposes a scale for measuring quality dimension in setting of e-government by an extensive review of literature from last 3 decades and by revising the SERVQUAL scale. The proposed scale is based on 25 items and six quality dimensions namely: reliability, responsiveness, ease of use\usability, website design\content, and security\ privacy. This research is fruitful for organizations to assess, measure, and improve the quality of service concerning the e-government services they offer.

Keywords-component; Service quality, E-service quality, E-government service quality, Determinant of quality, measurements of quality, Factors of E-service quality, Measurement of quality

I. INTRODUCTION

Although not marked curtly, technology and its advancements drive the societies and communities of the modern world. To keep up with the present age communication technology that provides prompt services, reinforcing technology needs to be implementing in the society. To this end, e-government, was introduced in the recent years, a terms that describes how to forte technology to boost the efficiency and quality of public administration in a government setting.

A prime challenge for the growing world is to develop e-government settings with quality of services. The eminent electronic government relies on its quality and usage [1]. Quality plays a leading role in maintaining trust between customer/citizen and government. The purpose of e-government in general, is to make information available online to citizens\users and allow them to communicate with the system by exchanging information. One of the most critical concerns of E-governments are to prioritize and invest in those quality factors that are of greater interest to citizens [2]. The governments need to improve the information communication technology for the assurance of reliable communication which fulfill the high quality service expectation of citizens/client[3].

(DOI Reference No:10.18293/SEKE2016-061)

The purpose of this research is to identify key elements of e-service quality dimensions that play an influential role in quality of e-services in setting of e-government. The basic purpose of identifying dimensions is to maintain the quality of service which meets the customer/citizen satisfaction level. Quality is a steering tool without which it is difficult for the government to provide effective projects that fulfills the need of users/citizens. The next session of the paper discuss the ways how the research has been conducted, the literature review of the e-service quality and e-government service quality is discussed, and on the basis of the studies the dimensions of the proposed instrument is presented.

II. METHOD OF STUDY

The purpose of the review is to identify the current issues that help the practitioners and the academicians to get the up to date information about the recent and relevant information about the current trends of e-government.

To conduct an extensive literature review based on systematic process with good clarity, completeness and conciseness, the systematic literature review (SLRs) research methodology is adopted in this study. Two types of SLR are conventional systematic literature review and Mapping Studies[4]. In Conventional Systematic Literature Review collection of literature is based on a specific research question and Mapping Studies is a process that finds and classifies the primary studies in a specific topic area. In this paper, Mapping Studies methodology is adopted for the sake of conducting systematic literature review since, in the context of this paper, primary studies are the key sources to find out the effective quality dimensions in domain of e-government.

A. Data Extraction

The search process involves the reading of articles from world known journals. The sources of studies are from the different electronic data base that includes, IEEE Xplore, ACM Digital Library, Science Direct, Springer Link, Engineering Link, Google scholar, Research Gate, Scopus. Different electronic resources are utilized that includes Asian Social sciences Journal Canada, International Journal for innovation education and research and journal of service marketing, Emerald etc. During search of the content it is ensure that no important data/ information is missing. The research involves in finding

the key dimensions for E government services. The key words used during the search includes, E-government, E-services, Dimension for e-services, dimensions for E-governemnt, E-services quality models, E-government Quality models. During the search, 195 papers were found from different journals and proceedings of conferences, out of which 70 were effective for further reading.

The universities of different countries are doing research on E services / E-govt services. In general the universities from United States are in the leading position. United states stands as the pioneers in providing test bench for studies of service quality by providing a test model SERVQUAL and it has been adopted by numerous authors as reference for their studies[5]. The universities of Greece , and Malaysia also provides different scales to evaluate e government quality named as E-govqual [6]. A multiple item scale for assessing e-government services quality tested by the online survey for different demographic values[7]. United Kingdom is also playing a vital role where also number of universities studying the e-services and e-government services dimensions. The University of Bath involved in research over data triangulation and web quality matrices[8]. In Asia, China is in the leading position as there also many universities involve in this area. Where, University of shanghai studied user satisfaction by assessment of E-govt service quality. University of Calicut and University of Kerala performed a comparative analysis of E-banking by using serv-qual model in public and private sector banks [9]. The universities of GCC region are also studying E govt service quality, but they are lacking a part. King Abdul Aziz university of KAE theoretically studied the satisfaction of end user by studying dimensions[10].

Sultan Qaboos University, Oman in 2015 analyzed trends of the quality dimensions in the context of evaluating e-government services [11]. Another research at Sultan Qaboos University in 2013 found that the Royal Oman Police were the most prominent department that had adopted E-services. The research involves focused group interviews and also a survey whose results based on around 800 questionnaires. The research was the first step for evaluating the E-government services in Oman but lacks on studying E-government services in different cities with different demographic values[12].

III. LITERATURE REVIEW

A. Service Quality

The scope of research reveals that studying services quality in different domains is growing rapidly. These domain can mainly divided in three parts, service quality , E-service quality, E-government service quality. Method of quantifying service quality and the dimensions of service quality has become considerable area in promoting e-government. A SERVQUAL model presented in mid 1980s is considered as the bench mark for researchers. The concept of service quality is new to researchers and not specific to any class [13].

SERVQUAL is a well-known model that represent ten service quality dimensions for measurement of quality:

responsiveness, competence, access, courtesy, communicating, creditability, security, understanding/ knowing the customer, and tangible[5]. Revised SERVQUAL model simplifies into five key dimensions which are reliability, assurance, tangibles, empathy, and responsiveness. This is a traditional model that is adopted by researchers in different domain as a reference frame in their studies. Parsuraman (1988) presented a 22 items based a comprehensive SERVQUAL model for assessing service quality in service and retail organizations[14].

B. E- Service Quality

Maintaining the quality of e-service is becoming principal for providing satisfactory services to the citizens/users. For measuring quality various researchers proposed various dimensions in different domains and context of applications. SITEQUAL was proposed to measure the perceived quality of the internet shopping site[15], Madhu & Madhu (2002) proposed 15 dimensions of e-services quality; performance, features, structure, aesthetics, reliability, storage capacity, Service ability, security and system integrity, trust, responsiveness, productive services, Web store policy, reputation, assurance and empathy[18], Wolfinger (2003) in united states presented a model ETAILQ for online retailing by identifying four key quality dimensions website design, fulfillment/reliability, privacy/security and customer services[19]. Santouridis (2012) examined the applicability of E-S-Qual and identified four dimensions: efficiency, fulfillment, system availability and privacy[30]. Janita (2013) explored service quality dimensions in B2B e-marketplaces and identified four dimensions: reliability, privacy, utility or the information, valued-add service [31]. Achchuthan (2014) in Sri Lanka developed an empirical model of service quality in terms of electricity services. The following dimensions, Tangibility, Empathy, Responsiveness, Reliability and Assurance are being utilized to conduct a study, In a result, 300 usable responses are collected from end users In the context of e-government, quality dimensions of e-service become important ingredients to measure the satisfaction level of users/citizens. In the last decade, many researchers conducted various researches to determine effective quality dimensions and measuring methods that influence the quality of e-services in e-government[6, 36-49]

C. Service Quality Dimensions for E-Government

Advancements in the information technology forced governments to adopt changes and provide e-services in government domains. In a larger context, e-services in e-government require more satisfaction of end user in term of services quality. The satisfaction level of end user can only assess by measurement of e-service quality dimensions. Abhichandani (2005) from united states firstly measure the e service quality dimension on government web sites. A total of 416 respondents of Los Angeles and Minneapolis evaluated the three different dimensions utility efficiency and customization. On the basis of the results a future frame has also suggested for other authors[37]. In 2006, Ibrahim conducted a survey by using e-Sq approach to study the e

service quality of UK banks. The study based on the 135 samples from UK banking customers to evaluate the perceived service quality on the basis seven dimensions , convenience/accuracy, accessibility/reliability, good queue management, personalization, friendly\responsiveness, customer service, and targeted customer service[50].

Arathy (2015) used SERVQUAL model to perform a comparative analysis of e-Banking in India. The dimensions used to evaluate the reliability, responsiveness, competence, access, communication, credibility, security and tangibility. The study suggests that service quality dimensions has strong impact on customer satisfaction[55]. A study is conducted in India where different Brand perception, responsiveness, merchandising, reliability, trust/security, website design and easy to use were evaluated and found that they are the important factors for customer satisfaction[56].

IV. PROPOSED INSTRUMENT

The proposed scale has six quality dimensions and 25-items. The scale is based on SERVQUAL scale and literature review from the last 3 decades. The table below shows that the dimensions of the proposed scale are frequently used by different authors in their studies.

After analysis of data presented in Table 1, reliability is the most important dimension which has been discussed by different authors in their studies. Security/privacy is also the major concern for authors whose frequency dramatically increases in the last five year. Web Site design/content is also an important dimension, that's why it is also the part of discussion of different authors. The frequency of website design and content significantly rise in between 2006 and 2010. The frequency of ease of use/ usability shows that it has been vastly studied by different authors in last decade. The dimensions of proposed scale are website design/content (6-Items), efficiency (3-Items), security/privacy (5-Items), ease of use\ usability (6-Items), responsiveness (5-Items), and reliability (3-Items).

TABLE I. PROPOSED INSTRUMENT

Reliability	Reliability is a commitment to ensure reliable, accurate and on-time services are provided by the e-government websites to its citizens/users.
Responsiveness	Responsiveness is a commitment to ensure availability of online e-government services and to help customers or citizens by providing prompt services when using the e-government websites.
Ease of Use/Usability	Ease of use represents having a friendly interface or environment between customer/citizens and government by owning a website that is user-friendly and accessible by anyone from anywhere.
Website Design/Content	Website Design/content refers to the functionality of website in terms of efficiency, visual appeal and useful content in an e-government websites.
Efficiency	Efficiency in an E-government website ensures that the services are up-to-date, efficient and satisfies the requirements of both customers/citizen and the government.
Security/Privacy	Security/privacy represents a trust which assures that the information of citizens/customers is safe, secure and safe from infringement in e-government websites.

V. DISCUSSION & CONCLUSION

The objective of this research is to investigate the quality dimension for measuring e-quality service so as to provide quality service to the users/citizens in domain of e-government. Through investigation of previous literature regarding quality dimensions and models, this research proposed a 25-items six dimensions quality scale in the context of e-government service quality which is based on the revised SERVQUAL. The proposed six dimensions are: reliability, responsiveness, ease of use/usability, website design/content, and security/ privacy. The proposed scale is valuable to countermeasure the factors which are influencing the quality of service in domain of e-government services.

ACKNOWLEDGMENT

The Research leading to these results has received Research Project Grant Funding from the Sultan Qaboos University of the Sultanate of Oman, Research Grant Agreement No [SR/EPS/INFS/14/01].

REFERENCES

- [1] S. F. H. Zaidi, *et al.*, "Development and Validation of a Framework for Assessing the Performance and Trust in E-Government Service," *International Journal of Applied Information Systems (IJ AIS)*, vol. 17, 2014.
- [2] T. H. AlBalushi and S. Ali, "Evaluation of the quality of E-government services: Quality trend analysis," in *Information and Communication Technology Research (ICTRC), 2015 International Conference on*, 2015, pp. 226-229.
- [3] A. Manoharan, "A three dimensional assessment of U.S. county e-government," *State and Local Government Review* vol. 45, pp. 153-162, 2013.
- [4] B. Kitchenham, *et al.*, "Systematic literature reviews in software engineering – A tertiary study," *Information and Software Technolog*, vol. 52, pp. 792–805, 2010.
- [5] A. Parasuraman, *et al.*, "A Conceptual Model of Service Quality and Its Implications for Future Research," *The Journal of Marketing*, vol. Vol. 49, No. 4 1985.
- [6] M. Alanezi, *et al.*, "A proposed instrument dimensions for measuring e-government service quality," *International Journal of u-and e-Service, Science and Technology Service*, vol. 3, pp. 1-18, 2010.
- [7] X. Papadomichelaki and G. Mentzas, "E-GovQual: A multiple-item scale for assessing e-government service quality," *Journal of Government Information Quartley*, vol. 29, pp. 98-109, 2012.
- [8] S. J. Barnes and R. T. Vidgen, "Data triangulation and web quality metrics: A case study in e-government," *Journal of Information & Management*, vol. 43, pp. 767-777, 2006.
- [9] C. Arathy and B. V. Pillai, "Customer satisfaction on e-banking services in public and private sector banks: a coparative analysis using SERVQUAL model," *International Journal of Business and Administration Research Review (IJBARR)*, vol. 2, pp. 106-111, 2015.
- [10] F. Al- Farsi and A. Basahel, "The sequence of electronic service quality on customer satisfaction: theoretical study," *International Journal for Innovation Education and Research*, vol. 2, pp. 10-24, 2014.
- [11] T. H. AlBalushi and S. Ali, "Quality Dimensions Trend Analysis in the Context of Evaluating E-government Services," *Research Journal of Applied Sciences, Engineering and Technology*, vol. 11, pp. 315-324, 2015.
- [12] S. K. Sharma, *et al.*, "Exploring Quality of E-Government Services in Oman," *Education, Business and Society: Contemporary Middle Eastern* vol. 6, pp. 87-100, 2013.

- [13] A. Parasuraman, et al., "A conceptual model of service quality and its implications for future research," *The Journal of Marketing*, vol. 49, pp. 41-50, 1985.
- [14] A. Parasuraman, et al., "SERVQUAL: a multiple-item scale for measuring consumer perception of service quality," *Journal of Retailing*, vol. 64, pp. 12-40, 1988.
- [15] B. Yoo and N. Donthu, "Developing a scale to measure the perceived quality of an Internet shopping site (SITEQUAL)," *Quarterly Journal of Electronic Commerce*, vol. 2, pp. 31-47, 2001.
- [16] E. T. Loiacono, et al., "WEBQUAL: A measure of website quality," *Marketing Educators Conference: Marketing Theory and Applications*, vol. 13, pp. 432-437, 2002.
- [17] E. T. Loiacono, et al., "WebQual Revisited: Predicting the intent to reuse a website," presented at the Eighth Americas Conference on Information Systems, 2002.
- [18] C. N. Madu and A. Madu, "Dimensions of e-quality," *International Journal of Quality and Reliability Management*, vol. 19, pp. 246-258, 2002.
- [19] M. Wolfinger and M. Gilly, "eTailQ: dimensionalizing, measuring and predicting e-tail quality," *Journal of Retailing*, vol. 79, pp. 183-198, 2003.
- [20] S. Cai and M. Jun, "Internet users' perceptions of on-line service quality: a comparison of online buyers and information searchers," *Managing Service Quality: An International Journal*, vol. 13, pp. 504 - 519, 2003.
- [21] M. Long and C. McMellon, "Exploring the determinants of retail service quality on the Internet," *Journal of Services Marketing*, vol. 18, pp. 78 - 90, 2004.
- [22] A. Parasuraman, et al., "E-S-QUAL a multiple-item scale for assessing electronic service quality," *Journal of Service Research*, vol. 7, pp. 213-233, 2005.
- [23] A. Caruana and M. T. Ewing, "The psychometric properties of eTail quality," *International Marketing Review*, vol. 23, pp. 353 - 370, 2006.
- [24] M. Fassnacht and I. Koese, "Quality of electronic services; conceptualizing and testing a hierarchical model," *Journal of Service Research*, vol. 9, pp. 19-37, 2006.
- [25] K. Heinonen, "The role of customer participation in creating e-service value," presented at the Conference Proceedings of frontier of e-business research, 2006.
- [26] H. H. Bauer, et al., "eTransQual: A transaction process-based approach for capturing service quality in online shopping," *Journal of Business Research*, vol. 59, pp. 866-875, 2006.
- [27] E. Cristobal, et al., "Perceived e-service quality (PeSQ)," *Managing Service Quality: An International Journal*, vol. 17, pp. 317 - 340, 2007.
- [28] C. Sohn and S. Tadisina, "Development of e-service quality measure for internet-based financial institutions," *Total Quality Management*, vol. 19, pp. 903-918, 2008.
- [29] S. Akinci, et al., "Re-assessment of E-S-Qual and E-RecS-Qual in a pure service setting," *Journal of Business Research*, vol. 63, pp. 232-240, 2010.
- [30] L. Santouridis, et al., "Using E-S-QUAL to measure internet service quality of e-commerce websites in Greece," *International Journal of Quality and Service Sciences*, vol. 4, pp. 86-98, 2012.
- [31] M. S. Janita and F. J. Miranda, "Exploring service quality dimensions in B2B e-marketplaces," *Journal of Electronic Commerce Research*, vol. 14, pp. 363-386, 2013.
- [32] Y. Li, "Main factors affecting the online service satisfaction an empirical study in china," *International Journal of Smart Home*, vol. 8, pp. 131-144, 2014.
- [33] S. Hussain, "Measuring quality of electronic service (e-service) in banking," *International Journal of Engineering Research and Applications*, vol. 4, pp. 350-359, 2014.
- [34] S. Achuthan, et al., "Service quality dimension of electricity services: evidence from electricity board in Sri Lanka," *Asian Social Science*, vol. 10, pp. 194-203, 2014.
- [35] H. Muhammad and G. I. Tanko, "Antecedents of e-service, quality, perceived value and moderating effect of e-satisfaction with e-loyalty in airline industries" *International Journal of Economics, Commerce and Management* vol. III, May, 2015.
- [36] A. Abanomy, et al., "E-government website accessibility: In-depth evaluation of Saudi Arabia and Oman," *Electronic Journal of e-Government*, vol. 3, pp. 99-106, 2005.
- [37] T. Abhichandani, et al., "EGOVSTAT: Toward a robust measure of e-government service satisfaction in transportation," in *International Conference on e-Government*, Ottawa, Canada, 2005, pp. 1-12.
- [38] K. Al-Gharbi and A. Al-Kindi, "E-government initiative in the sultanate of Oman: the case of Ubar," in *Knowledge and Technology Adoption, Diffusion, and Transfer: International Perspective*, A. H. S. Zolait, Ed., ed: IRMA International, 2012, pp. 73-77.
- [39] M. Alanezi, et al., "E-government service quality: a qualitative evaluation in the case of Saudi Arabia," vol. The Electronic Journal on Information Systems in Developing Countries pp. 1-20, 2012.
- [40] D. Bhattacharya, et al., "E-service quality model for Indian government portals: citizens' perspective," *Journal of Enterprise Information Management*, vol. 25, pp. 246-271, 2012.
- [41] N. M. Hein, "A study on evaluation of e-government service quality," *International Journal of Social, Management, Economics and Business Engineering* vol. 8, pp. 16-19, 2014.
- [42] G. Kaisara and S. Pather, "The e-government evaluation challenge-A South African batho pele-aligned service," *Government Information Quarterly*, vol. 28, pp. 211-221, 2011.
- [43] J.-S. C. Lin and P.-L. Hsieh, "Assessing the self-service technology encounters: development and validation of SSTQUAL scale" *Journal of Retailing*, vol. 87, pp. 194-206, 2011.
- [44] A. M. Al Khouri, "eGovernment strategies the case of the United Arab Emirates (UAE)," *European Journal of ePractice*, vol. September 2012, pp. 126-150, 2012.
- [45] B. Magoutas and G. Mentzas, "SALT: A semantic adaptive framework for monitoring citizen satisfaction from e-government services," *Expert Systems with Applications*, vol. 37, pp. 4292-4300, 2010.
- [46] I. H. Osman, et al., "COBRA framework to evaluate e-government services: A citizen-centric perspective" in *iGov Workshop*, Brunel University, West London, 2011.
- [47] M. Rehman, et al., "Factors influencing e-government adoption in Pakistan," *Transforming Government: People, Process and Policy*, vol. 6, pp. 258 - 282, 2012.
- [48] M. A. Shareef, et al., "An empirical investigation of electronic government service quality: from the Demand-side stakeholder perspective," *Total Quality Management*, vol. 26, pp. 339-354, 2013.
- [49] Y. M. Sheibani and E. Fariborzi, "E-government service in Iran: looking at citizen satisfaction," presented at the 2011 International Conference on Communication Engineering and Networks IPCSIT, Singapore, 2011.
- [50] E. E. Ibrahim, et al., "Customer perception of electronic service delivery in the UK retail banking sector," *International Journal of Bank Marketing*, vol. 24, pp. 475-493, 2006.
- [51] S. Rotchanakitumnuai, "Measuring e-government service value with the E-GOVQUAL-RISK model," *Business Process Management Journal*, vol. 14, pp. 724 - 737, 2008.
- [52] D. X. Ding, et al., "e-SELFQUAL: A scale for measuring online self-service quality," *Journal of Business Research*, vol. 64, pp. 508-515, 2011.
- [53] E. T. Loiacono and S. Deshpande, "WebQual and its relevance to users with visual disabilities," presented at the HCIB/HCII Switzerland, 2014.
- [54] M. Stiglingh, "A measuring instrument to evaluate e-service quality in revenue authority setting" *Public relations review*, vol. 40, pp. 216-22, 2014.
- [55] Arathy and B. V. Pillai, "Customer satisfaction on e banking services in public and private sector banks - a comparative analysis using servqual model," *IJBARR*, vol. 2, Jan- March, 2015.
- [56] D. V. Akshya Singh, Vandana Bharti, "An Examination of the Relationship between Service Quality Dimensions, Overall Internet Banking Service Quality and Customer Satisfaction," *International Journal of scientific research and management*, vol. 3, pp. 1978-1988, 2015.

A Systematic Mapping Study on Legacy System Modernization

Everton de Vargas Agilar
Computer Centre
University of Brasília
Brasília, Brazil
evertonagilar@unb.br

Rodrigo Bonifácio de Almeida
Computer Science Department
University of Brasília
Brasília, Brazil
rbonifacio@unb.br

Edna Dias Canedo
Faculty of Gama
University of Brasília
Brasília, Brazil
ednacanedo@unb.br

Abstract

Legacy system modernization has gained increasing attention from both researchers and practitioners, mainly due to the need of maintaining legacy systems towards business needs and technology advances. In this way, a set of techniques, tools and terms related to software modernization have been proposed— although they have not been consolidated yet. This hinders the characterization of real modernization scenarios according to the existing literature. This paper synthesizes the existing contributions related to software modernization by means of a mapping study that characterizes the main results in terms of proposed processes, techniques, and tools. As one of our main findings, we report a lack of empirical studies trying to understand the benefits of using the existing approaches for software modernization

Keywords Legacy Systems; Software Modernization; Mapping Studies in Software Engineering.

1 Introduction

Modernizing legacy systems takes place when traditional maintenance practices no longer meet the needs of the organizations [1, 3]. In such a scenario, the basic goal is to cut maintenance costs, to turn the legacy systems more flexible to change, and to prolong their usage in a production environment. From the standpoint of organizations, legacy systems correspond to the applications that support business operations in an institution and consolidate most of the corporate data [3].

In spite of being a theme that attracts growing attention, both in the academy and industry, we still lack a summarization of the main research contributions related to the modernization of legacy systems. That is, with the aim of

adequately describing real modernization scenarios in a specific institution, we realized the need to conduct a Mapping Study (MS) to characterize the modernization of legacy systems in the context of software maintenance— for the reason that a MS helps researchers to review and consolidate results from studies on a given subject [7, 9]. Therefore, the main contribution of this paper is to characterize software modernization according to the existing literature, discussing the related terms, classifying the related research contributions, and presenting the main reasons that motivate an effort of software modernization (also according to the literature).

Based on the results of our analysis, we found that most of the research contributions to the area are related to the managerial aspects of software modernization (55.88% of the total publications). Accordingly, there exists a lack of research contributions describing (and validating) techniques and tools to support software modernization. The remainder of the paper is organized as follows. Section 2 explains the research method and protocol we adopt to conduct the mapping study. Section 3 describes the main results of the mapping study, by characterizing the research contributions in the field. We conclude the paper presenting some final remarks in Section 4.

2 Research Method

The execution of a MS in Software Engineering has become an established practice that involves a well-defined set of activities [9]. This section describes the protocol used, according to the existing recommendations about how to conduct this kind of research in software engineering. The MS protocol is a plan that contains the basic procedures that should be used in the MS [9], which favors the reproduction of the mapping study by other researchers and diminishes the risk of bias as mentioned in [7]. The remainder of this section presents the research questions, the search strategy, and the criteria for including and excluding the publications.

2.1 Research Questions

The research questions aim at characterizing the modernization of legacy systems in the domain of software maintenance, by identifying the main contributions and studies found in the literature on the subject. The questions are as follows

- (RQ1) What characterizes the modernization of legacy systems according to the existing literature?
- (RQ2) What processes, techniques, and tools have been suggested in the literature to support modernization activities of legacy systems?
- (RQ3) What are the reasons that lead organizations to modernize their legacy systems?

2.2 Search Strategy

The search strategy consisted of a manual activity surveying publications provided in the main conferences and journals of the Software Engineering research area. This strategy, referenced in [7], was adopted because the terms related to software modernization have not been well defined yet, and thus this manual strategy would allow us to find relevant articles that might be ignored in the case we used an approach based on search strings in digital libraries.

Accordingly, our search strategy was organized to be run in three stages. A list with the research sources was produced for each stage in an empirical way. This strategy was supplemented by the “snowball technique” [7], aimed at finding new primary research sources through the analysis of the references of the articles we found. The research sources selected were:

- (a) Research sources in stage 1
 - ICSE – Intl. Conf. on Software Engineering
 - TSE – Transactions on Software Engineering
 - SPE – Software: Practice and Experience
 - IEEE Software
- (b) Research sources in stage 2
 - ICSM – Intl. Conf. on Software Maintenance
 - WCRE – Working Conf. on Reverse Engineering
 - CSMR – Software Evolution Week
- (c) Research sources in stage 3
 - ACM Digital Library
 - IEEE Xplore
 - SpringerLink
 - SEI Digital Library
 - Science Direct

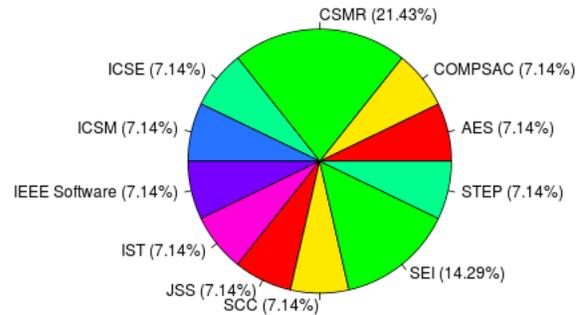


Figure 1. Publications by research sources

2.3 Inclusion and Exclusion Criteria

In order to select the more relevant primary studies, restrictions were set in place for inclusion and exclusion. As regards the inclusion criterion, we only considered publications that alluded to the software modernization theme either in the publication title or in the abstract, and works whose publication date fell between 1995 and 2015. This interval was set in place to yield the highest possible number of relevant publications. We excluded short papers (less than 4 pages in length) and works with less than 20 citations according to Google Scholar.

2.4 Screening of Publications

The selection procedure started with a manual search of the primary research sources that had been previously selected, according to the research protocol. This led to an initial list of 59 publications. This list was then reduced to 44 entries, following the use of a screening technique, as suggested by [9], which discards some publications that did not fit the criteria of the protocol. The final list of publications selected for our analysis can be found at the link <http://goo.gl/WwrGLY>.

Figure 1 summarizes this distribution, grouping by the main conferences and journals that published works related to the subject. The European Conference on Software Maintenance and Re-Engineering (CSMR) is responsible for the highest number of contributions (21.43%), followed by technical reports from the Software Engineering Institute (14.29%).

3 Results

This section describes the MS results obtained after the assessment of the selected publications. As regards the first question, we first analyzed the primary studies with the aim at characterizing *software modernization*. This aspect was then combined to produce the answer to the second question

(RQ2). Finally, w.r.t RQ3 we tackled the reasons that lead the organizations to modernize legacy systems, according to the surveyed literature.

3.1 Analysis of the First Research Question

To answer our first research question, an attempt was made to characterize the modernization of legacy systems in the domain of software maintenance. Therefore, as discussed in [1, 2, 3], modernization can be defined as *the evolution of systems towards new business requirements of the organizations, involving new functionalities, error correction, or technological updates*. In this sense, many theories have been suggested in the literature, as discussed below.

N. Weiderman et al. introduced a model for a software life cycle to describe the evolution of a production system [11]. According to this perspective, there are three distinct stages: maintenance, modernization, and replacement. Small modifications are made during the maintenance stage of a system, through small changes that aims at complying the system with new requirements or bug fixes. The changes with the greatest impact, such as important business requirements, changes in system architecture, or migration of a system to another platform, are done in the modernization stage. However, when the system becomes very resistant to evolution for some specific reason, it must be replaced. In this stage, the business needs of the organization are close related with the implementation efforts to meet these needs. Apart from introducing a life cycle model, Weiderman et al. also propose two approaches for software modernization: (a) white-box for understanding the internal structures of systems and (b) black-box for understanding the external interfaces of legacy systems.

K. Bennett et al. propose a model, entitled staged model, to also describe a system life cycle that assist to identify the main areas of the research on software modernization [1]. This model has 5 stages: initial development, evolution, servicing, phase-out, and close-down. Here, the concept of modernization entails the evolution stage and, differently to the model proposed by Weiderman et al., it is considered a maintenance activity, that can be further classified into 4 classes: adaptive, when there are changes in the software environment; perfective, for new user requirements; corrective, for bug fixes; and preventive, to avoid future problems.

J. Bisbal et al. propose a life cycle model that focuses on the evolutive activities as structured by the impact caused on the systems [3]. Thus, they are divided in wrapping, aimed at providing a new interface for the system's components, making them more accessible to other components; maintenance, for small adjustments and error correction; migration, aimed at moving the legacy system to a more flexible settings, though keeping the original data and functionali-

ties; and re-development, to completely re-write the applications.

It is possible to realize that, although these models use different terms to describe the life cycle of a system, they have many similarities. For instance, the meaning of *replacement* [11] is the same as *re-development* [3] and the meaning of *migration* [3] is equivalent to that of software *modernization* [11, 12]. In addition, the *wrapping stage* described by Bisbal et al. is similar to the *black-box modernization* technique according to Weiderman et al [11, 12].

In continuing with this appraisal, and due to the diversity of terms to describe the approaches for modernization, we answer the other research questions using the model and terminology proposed by Weiderman et al [11, 12]. Accordingly, we briefly introduce each stage in this evolutive model as follows

- **Maintenance** is the first stage in the life cycle of a system. It starts as soon as the system enter into production, being considered an iterative and incremental process through which small modifications are made in the system in a more localised way [1, 12]. However, as discussed in [11], these modifications only meet the needs of an organization for a certain period of time and eventually deteriorate the architecture of the systems.
- **Modernization** takes place when *maintenance* is not enough to keep the system up to date and aligned with the business goals. According to [1, 3, 12], modernization entails more significant changes, such as implementing a novel and relevant functional requirement, a modification on the software architecture, or a system migration to a new software platform. Therefore, as pointed in [1], modernization is more pervasive than maintenance, and this is one of the main aspects in their difference. Finally, as pointed out by [11], the work for modernization should preserve the data and the functionalities of a system, as it would otherwise be characterized as a replacement.
- **Replacement** (also known as *Big Bang* [3]) occurs usually when a legacy system becomes too resistant or inflexible to the work of modernization, there is a lack of documentation, or the cost of software maintenance can no longer be justified [1, 3, 12].

With this brief summary of the features of each stage in the life cycle of a system, the word-cloud of Figure 2 presents the 30 terms most cited in the abstracts of the primary sources selected. It should be noted that, from a technological perspective, there is a certain degree of interest on service-oriented computing in this figure.



Figure 2. Terms most frequently cited in selected publications

3.2 Analysis of the Second Research Question

To carry out the analysis and answer the second research question (RQ2), we split the publications into three groups, according to the classification proposed in [9]. First, regarding the *focus areas*, we identified three recurrent modernization strategies: black-box modernization, white-box modernization, and replacement. In this group we characterized all publications that propose a new modernization strategy.

We also classified the works by *contribution type*. According to this dimension, a contribution might be (a) *managerial*, for primary sources that describe some process, method, or methodology to manage the modernization process; (b) *technical*, for publications that propose a tool-based solution, such as frameworks, software libraries, and service buses, amongst others; and (c) *management and technical*, in the cases where the work describes a managerial and technical contribution. Finally, we classified the published works by considering the *research type* (as discussed in [13]).

Note in Figure 3 that 63.64% of the publications correspond to *solution proposal*, which suggests a lack of empirical studies trying to understand the benefits of using the existing approaches for software modernization. After concluding the classification of the primary sources, we then generated a bubble-plot reporting the frequencies and distributions of the approaches to modernize legacy systems as identified in the literature. Figure 4 presents that resulting distribution, synthesizing the studies identified in the literature on the subject, along with the gaps and opportunities for future research.

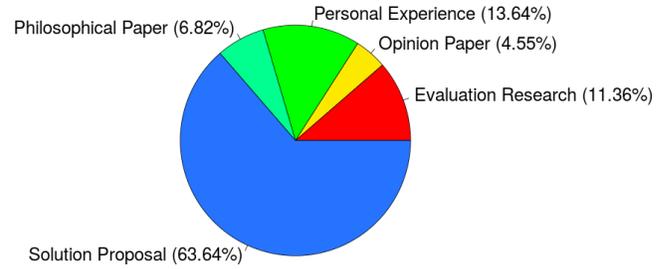


Figure 3. Types of research as reflected in the publications.

It is also possible to see that over 70% of the research work relates to the managerial aspects of the modernization activities. This scenario, according to [10], might be due to the fact that modernization projects have to be aligned with business, organizational, and technical perspectives. In this sense, a process of modernization can assist with several factors, such as: deciding whether developers should (a) continue to maintain the systems (since the costs still justify it); (b) proceed as a modernization effort (which might be the best option instead); or (c) conduct a replacement task when it is the only alternative.

Apart from that, when an organization has already decided that modernization is the only way to keep a competitive edge, one needs to decide which *strategies and techniques* will be the most appropriate for each situation. This way, as explained by [8], the decision on how a modernization project should be conducted requires a well-defined process, including modernization strategies, good practices, and the recommendations for an effective management of the project towards modernization. Several works have been proposed in this sense.

For example, Ransom et al. proposes a method for assisting in *system comprehension* [10] that should be undertaken as first activity of a modernization project. This method provides a guide to obtain the necessary information to understand a system, and thus allowing the architects to select a modernization strategy. Moreover, as regards the approaches used, 73.52% resorts to *white-box techniques*, with 23.52% proposing *black-box* ones. This suggests that reverse engineering techniques are promising and have been used to gain a grasp of the systems and to create tools to assist in the modernization process [5, 6, 10].

3.3 Analysis of the Third Research Question

According to the publications considered, it was possible to realize that there are four main reasons for modernizing a legacy systems, as described in the remaining of this section.

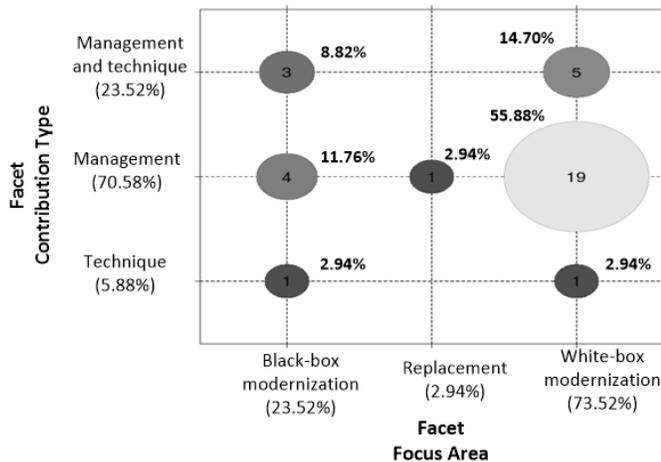


Figure 4. View of the studies identified

3.3.1 Lack of integration between systems

The demand for integrating existing systems is a growing factor in organizations. Software integration allows the automation of business process with improved resource management. However, according to [3, 5], many legacy systems have not been designed to facilitate software integration. This characteristic is considered one of the main reasons that motivates organizations to carry out a modernization effort, particularly towards the integration of business processes. In addition, there are several other benefits obtained as a result of a system integration effort, such as a reduction of duplicate implementation of business rules, the re-use of already-developed software solutions, and the reduction of development costs.

3.3.2 Reducing the maintenance costs

Reducing the maintenance costs of legacy systems is one of the major barriers to that organizations must overcome. According to [1, 3], legacy systems are those that are usually critical for the business and that present unjustifiable costs to be kept in operation. In [4], it is explained that the work of maintenance often monopolizes the efforts made by the organizations, as these activities, including error correction, adaptations, and general improvements consume from 50% to 70% of the budget related to a software effort. In addition, it has been pointed out that the lack of documentation and internal knowledge of the systems is one of the reasons for increasing software development costs, as well as the time spent in maintenance to correct failures in the software [3]. Therefore, as stated by [1], the dilemma faced is that, on one hand, the system is too valuable and a replacement can be too expensive to be considered. On the other hand, keeping a legacy system *up to date* might be too expensive, and thus it might be harder for an organization

to evolve the legacy systems so that they can fulfill the the business needs.

3.3.3 Lack of knowledge

As mentioned earlier, the lack of knowledge and the lack of legacy systems understanding is one of the reasons for a modernization project. According to [1, 3], understanding the design of a system is regarded as one of the requirements to implement the changes necessary by the organizations. It has been reported in the literature that a substantial part of the time needed to understand a legacy system lies in locating domain concepts in source code [1]. Thus, understanding the legacy systems is one of the central research problems in the literature, as discussed in [1]. For this reason, several research works have been proposed to identify alternatives for gaining a better understanding of the system, a vital component of any effort towards software evolution [1, 10].

3.3.4 Error proneness

Bennet also arguments that, due to the lack of updated documentation, modernization efforts are often made considering the source code as a reliable documentation [1]. Along with the issues in staff management, the systems can be affected by a lack of system and domain knowledge.

3.4 Threats to Validity

This study is limited to research in the literature to characterize the modernization of legacy systems in the context of maintenance software. Thus, the main threats to validity of this research is some possible bias in the procedures for selecting the publications. The research protocol consisted of manual searches in conferences and journals of Software Engineering—instead of using a search string, which is often applied in a MS. We believe that this decision helped us to obtain the most relevant articles for this study, in particular because the terms used to refer to *software modernization or evolution* of legacy systems are very wide (as discussed in Section 3.1). Our criteria led to 44 contributions published between 1995 and 2015, with at least 4 pages long and 20 citations according to Google Scholar. We also believe that this study can be reproduced by other researchers without the problems of publication bias. Of course, other studies might comprise different goals and research questions, and might also be more comprehensive. However, results or trends identified in this study should remain the same for the investigated period.

4 Final Remarks

Modernization of legacy systems has gained much attention in the last years, leading to a number of research contributions presenting new methods, techniques, and tools. Nevertheless, the lack of a suitable consolidation of these results hinders both researchers and practitioners to conduct their activities as well as to describe their findings and experiences using a common knowledge. In this paper we presented the results of a mapping study (MS) that consolidates the main contributions to the field. We found that the majority of the publications relies on a kind white-box modernization approach (often recovering the necessary information of legacy systems from the source code), which reinforces the need for reverse engineering tools. In addition, we also find that the managerial aspects are most relevant, which reinforces the idea of the importance of a good strategy of modernization. However, we found just a few studies reporting success experiences in modernizing legacy systems. Actually, most of the publications detail solutions that have been proposed without any practical evaluation. Finally, we found that there are four recurrent reasons reported in the literature to modernize a legacy system: the need for legacy systems integration, the need for improving software flexibility, the lack of knowledge about the system, and the error proneness for maintaining an existing system.

5 Acknowledgments

We would like to thank FAPDF Brazilian research funding agency for partially supporting this work.

References

- [1] BENNETT, K. Legacy systems: coping with success. *Software, IEEE* 12, 1 (1995), 19–23.
- [2] BIANCHI, A., CAIVANO, D., MARENGO, V., AND VISAGGIO, G. Iterative reengineering of legacy systems. *Software Engineering, IEEE Transactions on* 29, 3 (2003), 225–241.
- [3] BISBAL, J., LAWLESS, D., WU, B., AND GRIMSON, J. Legacy information systems: issues and directions. *IEEE Software* 16, 5 (Sep 1999), 103–111.
- [4] CANFORA, G., CIMITILE, A., DE LUCIA, A., AND DI LUCCA, G. A. Decomposing legacy programs: A first step towards migrating to client–server platforms. *Journal of Systems and Software* 54, 2 (2000), 99–110.
- [5] CHUNG, S., AN, J., AND DAVALOS, S. Service-oriented software reengineering: Sosr. In *System Sciences, 2007. HICSS 2007. 40th Annual Hawaii International Conference on* (Jan 2007), pp. 172c–172c.
- [6] FLEUREY, F., BRETON, E., BAUDRY, B., NICOLAS, A., AND JÉZÉQUEL, J.-M. Model-driven engineering for software migration in a large industrial context. In *Model Driven Engineering Languages and Systems*. Springer, 2007, pp. 482–497.
- [7] KITCHENHAM, B. Procedures for performing systematic reviews. *Keele, UK, Keele University* 33, 2004 (2004), 1–26.
- [8] LEWIS, G., MORRIS, E., AND SMITH, D. Service-oriented migration and reuse technique (smart). In *Software Technology and Engineering Practice, 2005. 13th IEEE International Workshop on* (2005), pp. 222–229.
- [9] PETERSEN, K., FELDT, R., MUJTABA, S., AND MATSSON, M. Systematic mapping studies in software engineering. In *Proceedings of the 12th International Conference on Evaluation and Assessment in Software Engineering* (Swinton, UK, UK, 2008), EASE’08, British Computer Society, pp. 68–77.
- [10] RANSOM, J., SOMERVILLE, I., AND WARREN, I. A method for assessing legacy systems for evolution. In *Software Maintenance and Reengineering, 1998. Proceedings of the Second Euromicro Conference on* (1998), IEEE, pp. 128–134.
- [11] SEACORD, R. C., PLAKOSH, D., AND LEWIS, G. A. *Modernizing Legacy Systems: Software Technologies, Engineering Process and Business Practices*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2003.
- [12] WEIDERMAN, N., SMITH, D., AND TILLEY, S. Approaches to legacy system evolution. Tech. Rep. CMU/SEI-97-TR-014, Software Engineering Institute, Carnegie Mellon University, Pittsburgh, PA, 1997.
- [13] WIERINGA, R., MAIDEN, N., MEAD, N., AND ROLLAND, C. Requirements engineering paper classification and evaluation criteria: a proposal and a discussion. *Requirements Engineering* 11, 1 (2006), 102–107.

Extending Architecture Description Languages With Exchangeable Component Behavior Languages

Robert Heim¹, Bernhard Rumpe^{1,2}, and Andreas Wortmann¹

¹ Software Engineering, RWTH Aachen University, Germany, www.se-rwth.de

² Fraunhofer FIT, Aachen, Germany, www.fit.fraunhofer.de

Abstract

Architecture description languages (ADLs) encapsulate domain concerns in components. Most ADLs enforce domain experts to use general purpose programming languages (GPLs) or an especially designed, fixed component behavior language. Domain-specific languages (DSLs), on the other hand, aim to reduce the conceptual gap between problem domain challenges and GPL solutions. Current ADLs focus on software engineering and disregard integration of domain-specific component behavior languages. We combine results from DSL-based software language engineering with component & connector ADLs to present a concept for the non-invasive and exchangeable integration of both. The concept is realized with the MontiArcAutomaton component & connector ADL. This liberates domain experts from using GPLs and facilitates their contribution.

1 Introduction

Engineering non-trivial software systems requires abstraction, domain expertise, and separation of concerns. Domain experts are rarely software experts. Enforcing their contribution via general-purpose programming languages (GPLs) introduces notational noise [1] and raises accidental complexities [2]. Instead, experts should be enabled to use the most appropriate domain-specific languages (DSLs). Their integration requires to separate domain concerns from integration concerns, while supporting to reuse participating DSLs in different contexts. Component & connector (C&C) architecture description languages (ADLs) [3] enable composition of software architectures from component models. Encapsulation of components empowers separation of concerns. Nonetheless, most ADLs require do-

main experts to contribute using GPLs or models of apriori fixed DSLs. The former gives rise to accidental complexities, the latter demands that domain experts use DSLs foreign to their domain. We present a concept for engineering C&C software architectures with exchangeable component behavior DSLs that facilitates contribution of domain experts. It relies on results from software language engineering. The contribution of this paper is a concept for language integration into C&C ADLs and its realization in MontiArcAutomaton [4] based on the MontiCore language workbench [5]. Sec. 2 motivates behavior language integration by example. Sec. 3 describes preliminaries. Afterwards, Sec. 4 presents the integration concept and Sec. 5 describes its realization. Sec. 6 discusses observations and Sec. 7 presents related work. Sec. 8 concludes.

2 Example

Consider a robotics company producing cleaning service robots. For better comprehension, reuse, separation of domain concerns, and translation into GPLs for multiple target platforms, the software architectures of the robots are modeled with a C&C ADL. The architecture for cleaning robots with a single arm to pick up garbage is depicted in Fig. 1. It relies on a garbage detector to locate garbage in its vicinity and uses a container checker to estimate whether it must be emptied. Based on their results, a central action controller derives the next action. The action is emitted via the controller's ports to a navigation component that realizes movement and to an arm component that operates the robot's arm. The behaviors of `GarbageDetector`, `ContainerChecker`, and `Navigation` are implemented in a GPL to interface robotics middleware (such as ROS [6]). The behavior of `ActionController` and `Arm` is modeled using DSLs: `ActionController` uses embedded automata (see Sec. 5), `Arm` a language to describe movement of a robot arm in terms of joint space locations [4].

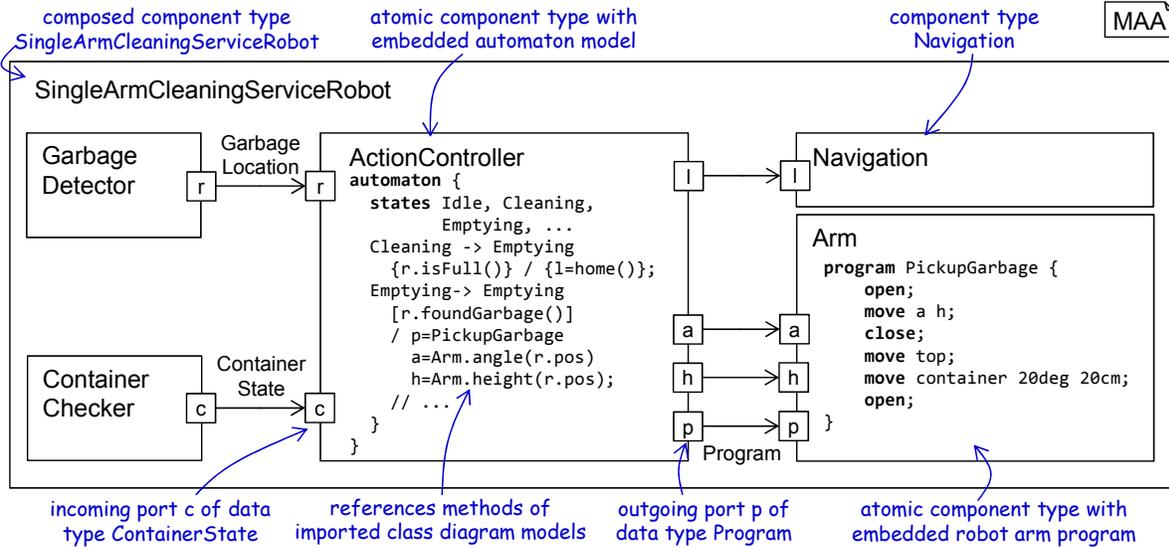


Figure 1. The software architecture for cleaning service robots with single arms comprises six components. The components **ActionController** and **Arm** encapsulate behavior in form of domain-specific language models.

The company’s robot behavior expert and its robot arm control expert can contribute models of these languages instead of dealing with the accidental complexities of programming. The embedding of the languages into components furthermore enables domain experts to contribute solutions without considering cross-cutting integration concerns. As they also are independent of the target platform’s GPL, the components can be transformed into artifacts for different GPLs to be reused with different platforms.

This illustrates how behavior language integration into a C&C ADL enables domain experts to use the most suitable languages to contribute component behavior and increases reuse of components with different platforms.

3 Preliminaries

MontiArcAutomaton [4] is an architecture modeling infrastructure for the model-driven engineering of C&C software architectures centered around the extensible MontiArcAutomaton ADL [7]. It supports translation of integrated components into GPL artifacts via compositional code generators [8].

With MontiArcAutomaton, modelers describe software systems as hierarchically composed components that interact via static connectors. Components encapsulate functionality behind interfaces of sets of typed, directed ports. Types of ports are modeled as class diagrams. Components either are atomic or composed: atomic components define input-output behavior via embedded behavior models or GPL be-

havior implementation artifacts. The input-output behavior of composed components is defined by the interaction of their subcomponents.

MontiArcAutomaton relies on the MontiCore [9, 5] language workbench. MontiCore languages are extended context-free grammars (CFG) with well-formedness rules implemented in Java. From a language’s grammar, MontiCore generates infrastructure to parse its models into abstract syntax trees (ASTs). It features comprehensive language composition mechanisms: *Language embedding* syntactically integrates languages to combine (parts of) different languages within one model at well-defined extension points. *Language aggregation* combines modeling languages to enable joint interpretation of their models, which remain in separate artifacts. *Language inheritance* enables to reuse the complete CFG of the parent language to add or extend its productions. Composition relies on symbol tables, which are data structures describing the essence of model elements free from the technical coercions of their ASTs. Every MontiCore language provides the infrastructure to create and to resolve symbols. This supports correct interpretation of names and well-formedness rule checking across integrated models. MontiCore processes models with *DSLTools* that reference the language they can process and its infrastructure.

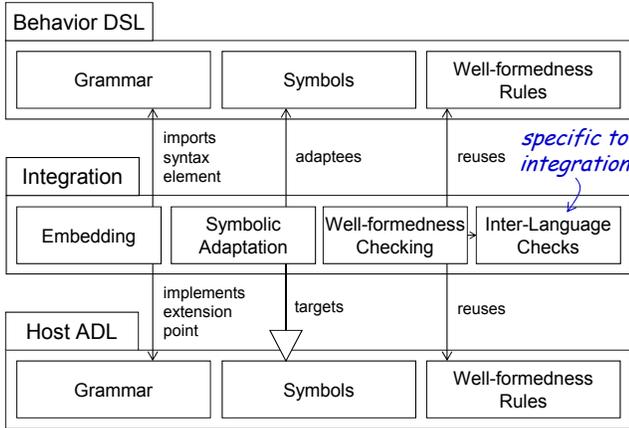


Figure 2. Behavior language integration relies on integration of syntax and static semantics.

4 A Concept for the Integration of Behavior Languages

For integration of behavior languages into MontiArcAutomaton, syntax and static semantics of both languages must be integrated. The latter include to reuse well-formedness rules of the embedded language and to capture changes in the meanings of references. Fig. 1 illustrates both: to parse the component `ActionController`, MontiArcAutomaton must be able to process the embedded automaton syntax as well. Furthermore, the meaning of references on transitions change: where automata models expect to operate on inputs and outputs of the automaton language, its integration should operate on ports of the surrounding component. Embedded models also are subject to (most) well-formedness rules of their stand-alone language. Integration also may entail new well-formedness rules: for embedded automata models, one might prohibit to define their own inputs and outputs in favor of ports.

Our approach to behavior language integration relies on MontiCore's language embedding, symbol adaptation, and well-formedness check reuse as depicted in Fig. 2. Embedding conditionally integrates parts of behavior languages into components, symbolic adaptation changes the interpretation of references, well-formedness checking reuses existing rules and integrates new rules. For syntactic embedding, language engineers must specify a mapping from behavior language elements to ADL language elements. Adaptation of the symbols expects that embedded languages operate on dedicated inputs and outputs. Otherwise, integration into components will hardly produce input-output behavior.

Overall, the goal is a black-box language integration in which model elements and well-formedness rules of stand-alone behavior modeling languages can be reused within

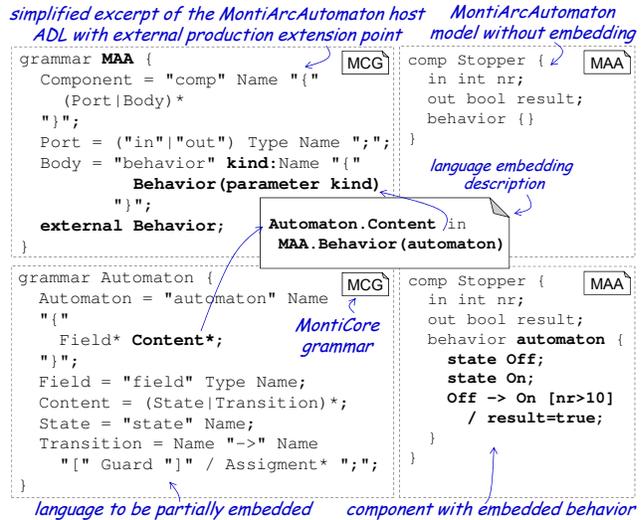


Figure 3. Embedding integrates parts of a behavior language into an extension point.

components to describe their input-output behavior. Thus, integration of behavior languages into MontiArcAutomaton entails the following requirements: **R1** The (partial) syntax of stand-alone behavior languages can be conditionally integrated into component models. **R2** The meaning of references used in integrated models can be changed. **R3** Selected well-formedness rules of the embedded language can be reused. **R4** Adding integration-specific well-formedness rules is possible. The latter should be as little as possible. Ideally, it is sufficient to map symbols between the languages and extra effort for inter-language rules is not necessary.

5 Integration into MontiArcAutomaton

Successful integration of behavior languages into components must combine the language's syntaxes, ensure a joint interpretation of references, and allow to reuse their well-formedness rules. The next sections present mechanisms for these aspects.

5.1 Syntactic Integration

The grammar of MontiArcAutomaton contains an *external* production [5] that acts as extension point for component behavior. A production of the behavior language is registered for this extension point with a specific keyword (e.g., `automaton` or `program`). Hence, whenever MontiArcAutomaton parses an integrated model meeting such keyword, infrastructure for processing the component behavior language's production is invoked. With this, integration acts

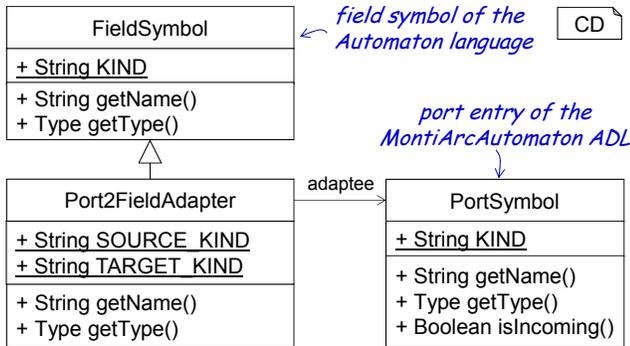


Figure 4. Adaptation between symbols allows to interpret references to behavior language inputs and outputs as references to component ports.

as a single grammar and produces a single combined AST. Fig. 3 illustrates language embedding and resulting models with an excerpt of the MontiArcAutomaton ADL (top left) and a simplified behavior language to define automata (bottom left). The MontiArcAutomaton ADL grammar contains the external `Behavior` production parametrized with the parameter `kind`. The latter is used to distinguish embedded languages and will become part of the concrete syntax. The top right of Fig. 3 shows a MontiArcAutomaton component model with two ports but without behavior. The `Automaton` grammar consists of the container production `Automaton` that contains multiple `Field` instances followed by multiple `Content` instances. The latter consist of arbitrary many states and transitions. Fields are either input data sources or output data sinks, states are names, and transitions consist of a source state name, a target state name, a guard, and assignments. The productions for guards and assignments are omitted as they are complex and do not contribute to embedding. The center of Fig. 3 depicts a mapping from production `Content` of `Automaton` to `Behavior` of MAA. The argument for this mapping is the keyword `automaton`, which is passed to be the `kind` of `MAA.Body` and helps MontiCore to distinguish which embedded production to select. It will also become a keyword in the concrete syntax of integrated models. Consequently, the integrated model (bottom right of Fig. 3) uses this keyword and arbitrary `Automaton.Content` elements for its `Behavior`. While this allows to embed partial syntax of behavior languages (R1) and to parse integrated models into combined ASTs, `Automaton` models expect `Field` instances for input and output, but should use ports when embedded. Also embedding does not integrate the well-formedness rules of the `Automaton` language. Both requires symbolic integration.

5.2 Integration of Symbols

The names used in a model to reference parts of the same or other models are symbolic references with certain meaning. For instance, the left-hand side of the assignment `result = true` of the transition depicted in Fig. 3 is only meaningful in an automaton if `result` is some form of `Field`. After embedding, MontiArcAutomaton prohibits fields in automata to avoid the underspecification from combining fields with ports. Consequently, the interpretation of `result` expecting to reference a field changes to referencing a port. Changing the symbolic interpretation of names in MontiArcAutomaton amounts to change the symbols it resolves when looking up a name. To this effect, MontiArcAutomaton provides infrastructure to add adapters between different symbols (such as fields and ports) to its symbol tables. Changing the interpretation of names in assignments from field references to port references requires that, whenever MontiArcAutomaton looks up a name expected to reference a field symbol, it returns a port symbol disguised as a field symbol instead (R2). With MontiCore, such change of interpretation is done by providing corresponding adapters (cf. Fig. 4). The `Port2Field` adapter registers to provide symbols of its `TARGET_KIND`, which corresponds to the `FieldSymbol` kind. Whenever MontiArcAutomaton tries to resolve a field symbol with a certain name, the adapter will return a port symbol of that name. This symbol than can be used to perform checks (e.g., whether `true` can be assigned to the port referenced by `result`). With the required adapters in place, all well-formedness rules of the `Automaton` language can be reused although there are only simulated `Field` symbols to check.

5.3 Integration Infrastructure

MontiArcAutomaton supports configuration of embedded productions, provision of adapters, and specification of well-formedness rules with a Groovy-based internal DSL [10]. Groovy allows to omit syntactic sugar (such as brackets around method arguments and dots between object expressions). As it further is compatible to Java, it can interface instances of the modeling language classes of MontiCore directly and allows to reuse the stand-alone infrastructure of behavior languages.

Listing 1 shows a model of the Groovy behavior configuration modeling language (GBC). This model first defines the condition and keyword `automaton` for embedding of the production `Automaton.Content` into the MontiArcAutomaton ADL (ll. 1-2). It also references the stand-alone DSLTool instance of the `Automaton` language (l. 3). From this, it retrieves the language's symbol table infrastructure and well-formedness rules (R3). Afterwards, it adds the integration-specific well-formedness rule

```

1 name "automaton"
2 behavior "Automaton.Content"
3 tool new AutomatonDSLTool()
4 coco new NoFieldsInEmbeddedModels()
5 adapter new Port2FieldAdapter()

```

Listing 1. GBC model for integrating the Automaton behavior language into MontiArcAutomaton.

NoFieldsInEmbeddedModels (R4) regarding prohibition of Field instances in embedded models (l. 4) and the Port2FieldAdapter depicted in Fig. 4 to interpret field references as port references. The data types of tool, coco, and adapter must correspond to the data types specified with the fluent interface of the class GBCBuilder that GBC operates on. Fig. 5 shows this class with its quintessential members and related data types. The GBCTool is a DSLTool that extends the MAATool, processes GBC models, constructs BehaviorConfiguration instances from these and parametrizes the MAATool with their information. Thus, using the GBCTool with corresponding GBC models allows to parse and check component models with integrated behavior languages.

6 Discussion

The mechanisms currently also allow to integrate arbitrary languages into the host ADL. Whether resulting combined models can produce input-output behavior is not checked yet. This would require means to designate productions of the behavior languages' grammars as input and output elements. Such a designation also would allow to generate parts of the adapters. Furthermore, the mechanisms could validate integration of the behavior languages' dynamic semantics with the messaging semantics of the MontiArcAutomaton ADL. As MontiCore languages codify dynamic semantics via code generators, this requires proper extension of MontiArcAutomaton's code generator composition framework [8]: explication of the semantics of produced artifacts enables the composition mechanism to reason over the semantic validity of specific language combinations. Embedding behavior language parts can introduce syntactical conflicts and MontiCore detects these at composition time and reports on these.

7 Related Work

The presented integration mechanisms are generalizable to other C&C ADLs. It requires the host ADLs to provide a well-defined extension points for component behavior and the behavior languages to specify input-output be-

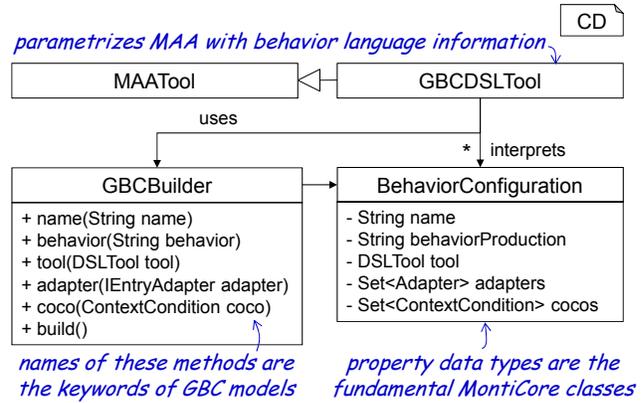


Figure 5. The methods of class GBCBuilder define the syntax of the GBC.

havior. However, most C&C ADLs disregard integration of component behavior DSLs [11]. To the best of our knowledge, similar integration is supported by AADL [12] and xADL [13] only. Both consider syntactic integration only, whereas MontiArcAutomaton also supports symbolic integration, reuse of well-formedness rules, and code generator composition [4] to translate integrated models into GPL artifacts.

To some extent, our approach relates to general language workbenches [14] as well. Most workbenches feature powerful language integration mechanisms, such as the abstract syntax embedding of Spoofox [15], and SugarJ [16]. These mechanisms are generic and their application to ADLs has yet to be defined for our purposes.

Our approach also relates to the byADL framework for ADL model-driven development [17]. The framework presents general meta model composition mechanisms operations providing great flexibility. Similar to general language workbenches, this freedom entails complexity and requires integrators to amass expertise in software language engineering. Our operations very specifically embed behavior into well-defined extension points of C&C ADL components and reduce the complexity for integrators.

8 Conclusion

We have presented a concept for syntactic and symbolic integration of behavior DSLs into components of a C&C ADL. It relies on well-defined extension points in the host ADLs abstract syntaxes allows to reuse abstract syntax and static semantics of behavior languages. Such integration facilities participation of domain experts as it enables to use the most appropriate modeling languages to describe component behavior.

References

- [1] D. S. Wile, "Supporting the DSL Spectrum," *Computing and Information Technology*, 2001.
- [2] R. France and B. Rumpe, "Model-Driven Development of Complex Software: A Research Roadmap," in *Future of Software Engineering 2007 at ICSE.*, 2007.
- [3] N. Medvidovic and R. Taylor, "A Classification and Comparison Framework for Software Architecture Description Languages," *IEEE Transactions on Software Engineering*, 2000.
- [4] J. O. Ringert, A. Roth, B. Rumpe, and A. Wortmann, "Language and Code Generator Composition for Model-Driven Engineering of Robotics Component & Connector Systems," *Journal of Software Engineering for Robotics*, 2015.
- [5] H. Krahn, B. Rumpe, and S. Völkel, "Monticore: a framework for compositional development of domain specific languages," in *International Journal on Software Tools for Technology Transfer (STTT)*, 2010.
- [6] M. Quigley, B. Gerkey, K. Conley, J. Faust, T. Foote, J. Leibs, E. Berger, R. Wheeler, and A. Ng, "ROS: an open-source Robot Operating System," in *ICRA Workshop on Open Source Software*, 2009.
- [7] J. O. Ringert, B. Rumpe, and A. Wortmann, *Architecture and Behavior Modeling of Cyber-Physical Systems with MontiArcAutomaton*, ser. Aachener Informatik-Berichte, Software Engineering. Shaker Verlag, 2014, no. 20.
- [8] J. O. Ringert, A. Roth, B. Rumpe, and A. Wortmann, "Code Generator Composition for Model-Driven Engineering of Robotics Component & Connector Systems," in *1st International Workshop on Model-Driven Robot Software Engineering (MORSE 2014)*, ser. CEUR Workshop Proceedings, vol. 1319, York, Great Britain, July 2014, pp. 66 – 77.
- [9] H. Krahn, B. Rumpe, and S. Völkel, "MontiCore: Modular Development of Textual Domain Specific Languages," in *Proceedings of Tools Europe*, 2008.
- [10] M. Fowler, *Domain-Specific Languages*. Addison-Wesley Professional, 2010.
- [11] I. Malavolta, P. Lago, H. Muccini, P. Pelliccione, and A. Tang, "What Industry Needs from Architectural Languages: A Survey," *IEEE Transactions on Software Engineering*, 2013.
- [12] P. H. Feiler and D. P. Gluch, *Model-Based Engineering with AADL: An Introduction to the SAE Architecture Analysis & Design Language*. Addison-Wesley, 2012.
- [13] L. Naslavsky, H. Z. Dias, H. Ziv, and D. Richardson, "Extending xADL with Statechart Behavioral Specification," in *Third Workshop on Architecting Dependable Systems (WADS)*, 2004.
- [14] S. Erdweg, T. van der Storm, M. Vliet, M. Boersma, R. Bosman, W. R. Cook, A. Gerritsen, A. Hulshout, S. Kelly, A. Loh, G. D. Konat, P. J. Molina, M. Palatnik, R. Pohjonen, E. Schindler, K. Schindler, R. Solmi, V. A. Vergu, E. Visser, K. van der Vlist, G. H. Wachsmuth, and J. van der Woning, "The State of the Art in Language Workbenches," in *Software Language Engineering*. Springer International Publishing, 2013.
- [15] G. H. Wachsmuth, G. D. Konat, and E. Visser, "Language design with the spoofax language workbench," *Software, IEEE*, vol. 31, no. 5, pp. 35–43, 2014.
- [16] S. Erdweg, L. C. Kats, T. Rendel, C. Kästner, K. Ostermann, and E. Visser, "Library-based Model-driven Software Development with SugarJ," in *Proceedings of the ACM international conference companion on Object oriented programming systems languages and applications companion*. ACM, 2011, pp. 17–18.
- [17] D. Di Ruscio, I. Malavolta, H. Muccini, P. Pelliccione, and A. Pierantonio, "Developing Next Generation ADLs Through MDE Techniques," in *Proceedings of the 32nd ACM/IEEE International Conference on Software Engineering-Volume 1*. ACM, 2010, pp. 85–94.

The Software Architecture Mapping Framework for Managing Architectural Knowledge

Sébastien Adam, Alain Abran

Department of Software and IT Engineering
École de technologie supérieure
Montréal, Canada

Abstract—Within a software architecture design (SAD) project, designers deal with software design artifacts (SDAs) such as scenarios, patterns, and tactics. Each SDA has its unique issues and related architectural knowledge (AK) that may threaten the success of a project. This paper introduces the Software Architecture Mapping (SAM) framework to manage AK and associated issues by using finer-grained SDAs and networks of weighted arguments. These networks of data may be used to produce quantitative information in multi-dimensional views to facilitate the identification of critical SDAs and issues in a project. This paper illustrates how the SAM framework has been used to manage AK related to the template method (TM) design pattern in the context of an academic case study.

Keywords: *architectural knowledge (AK), architectural knowledge management (AKM), decision support systems, multi-dimension analysis, software design artifact (SDA), software architecture design (SAD), software architecture mapping (SAM), software structures map (SSM)*

I. INTRODUCTION

During development of a software system, development teams deal with numerous software design artifacts (SDAs) such as scenarios, design patterns, and procedures. Each SDA can be characterized using a set of related SDAs and issues that are factors of influence that may threaten the success of a project. For instance, a design pattern [1] is an SDA characterized by a rationale, a solution, plausible consequences, and trade-offs that need to be considered for implementation. SDAs and related issues are assets of architectural knowledge (AK) that embody decisions and trade-offs applied during the project.

For development teams to be efficient, the SDAs and related AK must be managed and shared in an efficient fashion. Indeed, designers must evaluate how the most influential SDAs impact the capacity of the system to satisfy stakeholder needs. Insufficient details about these SDAs and their relationships may lead teams to inappropriate or suboptimal decisions. Several approaches propose a process or a technique aimed at managing SDAs [1,2, 3, 5, 7, 8, 10]. These approaches usually focus on a subset of the SDAs involved in the design process and on a specific development perspective that is part of the process. However, there is a lack of studies that support a multi-dimensional view of the AK database and a methodical treatment of the SDAs and related AK (i.e., the related SDAs, issues, and arguments that influence the activities and dimensions of a software development project).

Designer expertise and experience remains the key element for identifying the critical factors of a project and their appropriate solution. This is true at different stages of the development process (analysis, design, and implementation) and for different project aspects such as budget, quality, and schedule. AK should be managed in an integrated and systematic manner to enable the development of decision support systems that: 1) offer support to identify, describe, and analyze relevant SDAs, issues, and arguments; 2) relate SDAs to their factors of influence; and 3) keep track of the adopted arguments and resolved issues.

This paper proposes the Software Architecture Mapping (SAM) framework to manage AK and support multi-dimensional analysis of the AK database. The proposed AK model defines the following concepts: SDA, issue, software structures map (SSM), factor, argument, argumentation, and view. The SAM framework supports a two-phase approach for identifying, describing, and analyzing critical factors related to SDAs for a given project. The first phase is the assets creation phase, which aims at classifying the SDAs into one or more SSM and describing the related AK in the form of issues and interrelated arguments. The second phase is the assets consumption phase where the AK is used to provide views that facilitates identification of a project's critical factors.

The SAM framework has been applied in industrial contexts (software cockpit design and Web engineering) and academic contexts (catalogs of styles, patterns, and tactics, undergraduate design courses, and Web engineering) to evaluate its technical feasibility and usability, especially for novice designers. In addition, a requirements self-assessment has been conducted using the requirements for AK management proposed in the literature (e.g., architectural documentation rules [2]). As an example of SDA, this paper presents the template method (TM) design pattern [1] and related factors, which are analyzed and ranked to produce multi-dimensional views that highlight the critical factors of the case study. The contributions of this paper include: 1) reusable definitions of the AK model's constituents based on description formats for the SDAs, SSMs, issues, arguments, and argumentations; 2) a systematic method for executing a multi-dimensional analysis of the factors; 3) a flexible method to transform argumentations to multi-dimensional views.

The paper is organized as follows. Section II presents an overview of the SAM framework. Section III and Section IV illustrate the two phases of the proposed approach using a case study realized in the context of an undergraduate course. Section V presents related works and section VI the conclusions.

II. OVERVIEW OF THE PROPOSED SOFTWARE ARCHITECTURE MAPPING (SAM) FRAMEWORK

The SAM framework [11, 12] has been proposed to support software architecture design (SAD) and architecture knowledge management (AKM). The framework manages the AK defined by existing methods, models, and description templates from the literature on SAD and AKM [1, 2, 3, 7] (i.e., constraints, requirements, quality attributes, scenarios, concerns, rationale, styles, tactics, patterns, situational factors, assumptions, risks, components and connectors, fragments, viewpoints, views, procedures, metrics, and domain objects).

Figure I presents an overview of the SAM framework. The blank shapes represent the framework and the four basic concepts that constitute its reference model (i.e., the software design artifacts (SDAs), software structures map (SSM), argument, and view). The colored shapes are concepts of the Attribute-Driven Design (ADD) method [3].

The SAM framework defines two phases of knowledge processing: 1) asset creation is performed by a knowledge engineer, i.e., a software designer tasked with the creation of assets (i.e., SDAs, SSMs, arguments, and views); 2) asset consumption is performed by software designers that use the AK in the reusable assets of their projects. The asset creation phase elicits the factors that constitute the AK, followed by the asset consumption phase which analyzes these factors in order to create multi-dimensional views that enables identification of important factors of a software project. Each phase is independent. The results of the creation phase may be used for multiple executions of the consumption phase. Asset creation is organized into three steps: 1) identifying SDAs and related activities; 2) eliciting issues and impacted dimensions; and 3) describing arguments. The analysis phase is also divided into three steps: 1) selecting factors and building generic views for the SDAs under analysis; 2) ranking factors according to the context of the project and generating contextual views; and 3) identifying important factors of the project using the contextual views.

The two starting points in Figure I illustrate two ways to use the SAM framework. First, the SAM process may be used to acquire and share knowledge extracted from descriptions of styles, tactics, design patterns, and design decisions. Then, the resulting design knowledge base (i.e., SDAs and SSMs) may be used to support the SAD. At particular decision points in the design process, such as selection of a pattern, the software designers may use the SSMs of styles, tactics, patterns, or decisions as checklists of SDAs to elicit issues, describe arguments, and create views. For a specific decision point, an SSM may record the general, contextual, and design knowledge, and the arguments may record the reasoning, as proposed in the literature.

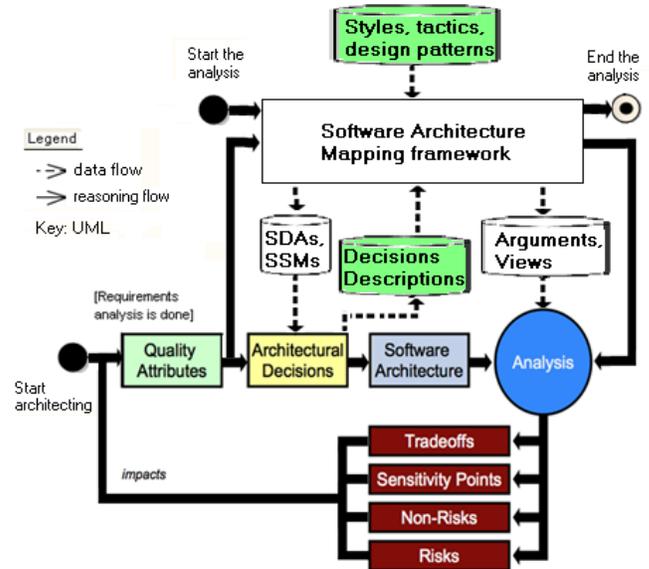


Figure I Overview of the SAM framework

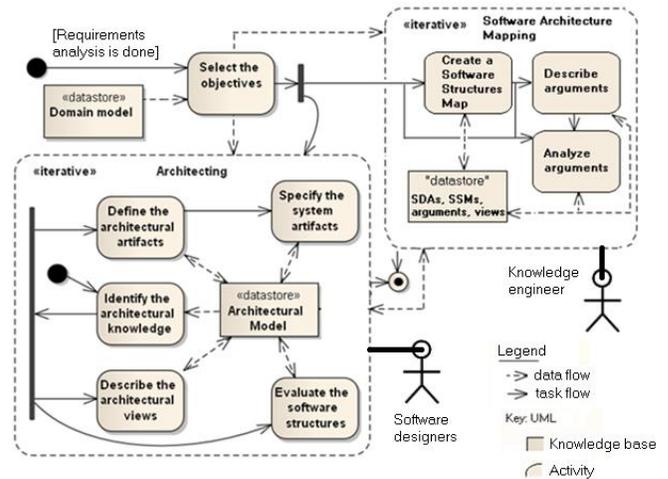


Figure II Overview of the SAM process

A. Process and roles of the proposed SAM framework

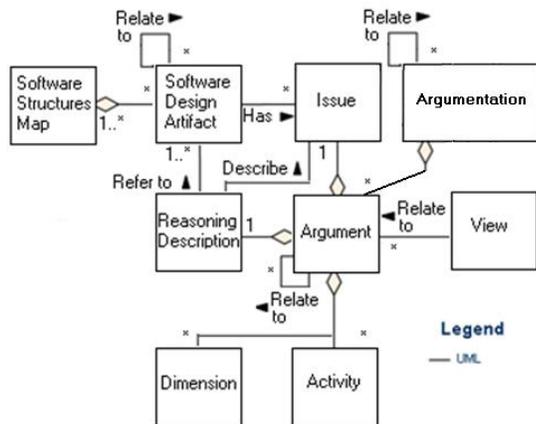
The proposed SAM process aims at managing the SDAs, SSMs, issues, and arguments related to a design. Figure II presents the three activities of the SAM process (i.e., create an SSM, describe arguments, and analyze arguments) and the task flow and data flow between the SAM process and the SAD. The activity “create an SSM” aims at 1) identifying the finer-grained SDAs related to either a decision point or the description of a style, pattern, or tactic, and 2) classifying these SDAs into a matrix of traceability. The activity “describe arguments” aims at 1) eliciting the issues related to the SDAs used and 2) reasoning about the arguments related to the issues. The activity “analyze arguments” aims at inferring the order of treatment of the arguments and issues based on the rankings and views of the AK repository created during the analysis.

B. The proposed architectural knowledge model

The proposed AK model is based on our previous work [11, 12], case studies, and controlled experiments applying the SAM framework. The AK model was developed by addressing the requirements and conclusions in the literature on methods, models, and tools for SAD and AKM. The model was designed to meet the following requirements: 1) capture rationale, constraints, design decisions, and related explications and quantifications about how they impact objectives; 2) reduce the possibility of expressing similar concepts with different terms; 3) take into account all activities and SDAs from the literature on SDA and AKM; 4) support personalization for context-specific SAD and AKM; 5) capture the SDAs and issues related to specific decision points; 6) capture the relationships between SDAs; 7) provide multiple perspectives for managing the AK repository; 8) support an integrated approach to SAD and AKM; 9) capture the AK from textual catalogs; 10) support selection and comparison of SDAs; and 11) support the evaluation of the SDAs and the consequences of applying each of them.

Figure III presents the concepts of the AK model for the SAM framework. AKM aims at sharing AK explicitly in a manner that supports AK evolution over time along with the architectures and their implementations. From our point of view, the SDAs and the arguments about their utilization constitute the explicit AK that relates to both the SAD and AKM. Each SDA has some related SDAs and issues. The SAM framework proposes to use: 1) the software structures map (SSM) for structuring the SDAs, 2) the argument for describing the issues and impacts related to the SDAs, 3) the view for analyzing the impact of arguments on dimensions and activities of SAD, and 4) the argumentation for structuring knowledge. The SDAs describe the general context and design knowledge, and the arguments describe the reasoning. The following sections describe the concepts of the AK model.

Figure III The AK model of the SAM framework



The SAM framework defines a structure of software design artifacts (SDAs) for AKM. This structure of SDAs is the basic concept supporting the proposed approach. Many SDAs and relationships between them are described in the literature. An SDA may be but is not limited to: a tactic, a quality attribute scenario, a measure, a design pattern, a style, or any input or outcome of the SAD [12].

Definition: An SDA is any conceptual artifact that 1) provides design knowledge about the problem or solution spaces of a software design, and 2) corresponds to the identification heuristics of the SAM framework.

An SDA is either elementary or composite. The proposed definition is that an elementary SDA does not require the utilization of another SDA in the design solution, while a composite SDA does require the utilization of another SDA from the solution space where it is being used. For example, a tactic is an elementary SDA as proposed in [3], while a design pattern and a style are composite SDAs [1, 2]. The tactics described in [3] require no SDA from the solution space. The TM design pattern requires the utilization of the polymorphism tactic [1]. An SDA may have one or more applications, resulting in multiple descriptions of tactics [3], design patterns [1], and styles [2].

Definition: A software structures map (SSM) is a matrix of traceability that records a set of SDAs used either at a particular decision point during the SAD or in the descriptions of styles, tactics, and patterns.

The SSM is an instantiation of the classification scheme (CS) of the SAM framework [12]. The CS uses a matrix where the columns represent the interrogatives (why, when, what, which, how, and where) and the rows represent the activities of the SAD. The SDAs of the following activities occupy the row labels: select the objectives, identify the knowledge, and define, specify, describe, and evaluate designs. An SSM is managed as part of the AK. The SAM framework relies on the knowledge base of SDAs and SSMs for supporting the SAD. Related work [12] presents the table format used for representing an SSM. Each interrogative regroups only the SDAs classified into the corresponding column of the SSM. The SDA type gives the corresponding line of the SSM.

Definition: An issue describes a problem that occurs by introducing an SDA into a system being developed. One or more issues may be elicited for every change to a system.

The SAM framework defines a specific format to describe the issues. An issue description is composed of an SDA (subject), a verb, and a complement. The SAM framework proposes a list of verbs used for describing the issues [11]. The verbs capture abstractions that provide additional data about the issues, and express something that alters the meaning of the issue descriptions. Verbs support change from ad-hoc issue descriptions to predicate-issue structures. The verbs are used as a mean to facilitate the elicitation of issues and provide an issue description format.

Definition: A factor is an essential element for analysing how an SDA such as a design pattern, a tactic, or a style may impact a software design.

The SAM framework applies a multi-dimensional analysis using sets of factors that influence software engineering. A factor may be an SDA, issue, claim, reasoning, activity, or dimension. Table I presents the names and the descriptions of the six proposed factors.

TABLE I. FACTORS THAT CONSTITUTE THE ARGUMENTS

Name	Description
SDA	Software design artifact being examined
Issue	Problem that occurs by using or not using an SDA
Claim	Solution that occurs by using an SDA
Reasoning	Reasoning description about an issue or a solution
Activity	Set of cohesive development tasks
Dimension	Perspective on a set of evaluation results

TABLE II. THE PROPOSED ARGUMENTATION DESCRIPTION FORMAT

Argument: issue or claim, reasoning, and scope of the argumentation.
Reasons: arguments that support the claim.
Rebuttals: arguments that establish the falsity of the claim.
Alleviations: arguments that reduce the intensity of the claim.

Definition: An argument is a reasoned attempt to convince the audience to accept a point of view about an issue or a claim. An argument is an aggregation of factors, including at least one or more SDAs, one issue or one claim, and one reasoning description. The argument’s scope may refer to impacted dimensions and activities.

A reasoning description describes the relationships between a set of factors; for example, between two SDAs – software designer and hook operation. The following reasoning description refers to two SDAs (software designer and object-oriented paradigm), an issue, and two dimensions (quality and people): “Using an object-oriented paradigm requires skills, expertise, and knowledge. The software designers do not master the object-oriented paradigm. This issue may impact the software quality and the software designers’ commitment.” The argument’s scope refers to activities and dimensions strengthened (+) or weakened (-) by the argument. The activities are inferred from the activities related to the SDAs identified in the argument’s reasoning. The dimensions are inferred from the dimensions impacted by the issue that prompted the argument. An argument is related to a dimension if there is a suspicion that the issue may produce variation (+ or -) of a dimension evaluation result.

Definition: An argumentation relates a set of arguments that describe how some SDAs create or resolve issues. Table II presents the argumentation description format. An argument provides the argumentation’s claim, reasoning, and scope. Reasons, rebuttals, and alleviations are connection points. Reasons are arguments that support the claim. Rebuttals are counter-arguments for the claim. Alleviations are arguments that affect the claim.

Definition: A view is a matrix that puts into perspective a set of ranked arguments. A view analyzes an argument’s impact on a design. In the SAM framework, the rows are labeled with activities such as designing, implementing, and managing, and the columns are labeled with dimensions such as functions, people, and quality.

The rankings of activities, dimensions, and arguments generate contextual views that are subjective and quantified. A views is used to identify critical factors of the project, which corresponds to the cell of a view that has a higher value. Cells are prioritized based on their values.

Cells with the highest priority (i.e., with a priority of 1) are used for reasoning further about factors that relate to the cell in order to nullify or reduce its value. Then, after these critical factors are addressed, the ranking are adjusted. The adjusted rankings provide new priorities. The analysis technique iterates these steps (i.e., identifying flaws and taking actions accordingly) until the user is satisfied with the values in the views (i.e., specific threshold values are attained).

III. ASSET CREATION PHASE

Section III illustrates the AK model’s concepts and the phases of the SAM framework using the TM design pattern [1] (see [11] for more details).

A. Eliciting SDAs and related activities

Table III presents some of the SDAs identified from analysis of various TM descriptions given in the literature. Each SDA has a description and is classified under a specific type. We used the classification scheme and SDA types proposed in [12] as a means to facilitate elicitation of SDAs and constrain their interpretations.

TABLE III. DISTINGUISHING SDAs OF THE TM PATTERN

Software design artifact (SDA)			Act.
Id	Type	Description	
Ra1	Rationale	Define an algorithm, defer steps to subclasses	D
Pr1	Property	Object-oriented paradigm	AD
Pr2	Property	Reusability	AD
Pr3	Property	Extensibility	AD
Be1	Behavior	Template method calls primitive operations	DI
Op1	Operational	Define an abstract base class	DI
Op4	Operational	Define a template method	DI
Op5	Operational	Define a concrete child class	DI
Op8	Operational	Declare protected primitives operations	DI
St1	Structure	Abstract class	I
St2	Structure	Concrete class	I
Ro1	Role	Subclass writers	M
SF1	Situational	Multiple kinds of primitive operations	ADIM
Co1	Convention	Naming convention	IM

It is important to address each SDA during the activities that produce the most beneficial influences. An activity is a set of cohesive tasks intended to contribute to the achievement of a common goal. Table IV classifies each SDA of the TM pattern based on these criteria. We considered four important activities: architecting (A), designing (D), implementing (I), and managing (M) [11].

B. Eliciting issues

We analyzed the TM pattern to identify issues that may hinder its usage. Table IV lists some issues and the related SDAs. Due to lack of space, we present only a few of the numerous issues identified. Each SDA may solve or engender one or more issues. For example, the extensibility property (SDA) may not be well defined for a module (issue). Also, to lighten the responsibility of the subclass writers, the template method calls the primitive operations (SDA). Uncontrolled calls to primitive operations (issue) may cause problems. To address this issue, the pattern declares protected primitive operations (SDA). Our approach was to use a semi-formal argument format to describe the issues.

C. Describing arguments and impacted dimensions

One important objective of the asset creation phase is to describe arguments to use during the consumption phase to estimate the impact of each issue, which may differ depending on the context of use of an SDA. Argumentation is concerned with reasoning in the presence of imperfect knowledge by eliciting arguments for exploring issues rather than eradicating them [4]. In our approach, the argumentation was geared towards quantifying the impact of the factors on project dimensions and activities. The project dimensions we considered were adapted from [6] (also see [11]): Functions (F), Quality (Q), People (P), Budget (B), and Schedule (S).

Table VI presents some of the arguments we elicited to establish how each issue of the TM pattern impacts project dimensions (F, Q, P, B, S) and activities (M, A, D, I). For example, the argument (Arg1) predicts positive impacts on the functional dimension (F+) by declaring a final method. One reason is that a final method cannot be overridden. One rebuttal or reservation is that it is possible to hack the final mechanism (Arg7). The argument refers to SDAs (e.g., SDA OP7) that may concern both design (D) and implementation (I) activities. The prevision was not weighted during the elicitation step because the elicited arguments were not project-specific. They can be reused among projects with other situational factors. The arguments are weighted during the analysis phase where a specific project is analyzed.

IV. ASSET CONSUMPTION PHASE

During the asset consumption phase, we used the factors elicited in the creation phase to engender multi-dimensional views for assessing the impact of factors in different contexts. It is a three step phase. The planning step selects factors and builds generic views of networked arguments related to these factors. The execution step ranks factors according to the specific context of the project and generates weighted views. These contextual views are then used to identify critical factors addressed by designers.

A. Selecting factors and building generic views

The TM pattern was selected as the SDA for analysis. Due to lack of space four activities (M, A, D, I) and five dimensions (F, Q, P, B, S) were considered as factors, and only some of the arguments related to the TM. By selecting activities and dimensions we obtained a generic multi-dimensional view of the TM arguments that relate to the factors under analysis. Table VII presents the view obtained from the arguments described in Table VI.

B. Ranking arguments, activities, and dimensions

We used absolute ranking (H: high, M: medium, L: low and X: not relevant) for prioritizing the factors. As a first step, a work team evaluated how much each activity and dimension was relevant to the project. The weighting of activities and dimensions may be different depending on the project's context and nature. These rankings were used for filtering the arguments that were then further analyzed from the multi-dimensional view of Table VII. In addition, the values of the rankings were used for multiplying the weights of the arguments.

TABLE IV. ISSUES RELATED TO THE TM

SDA	Issue description
Co1	The naming convention is not well defined
Be1	The template method behavior is subject to change
Op1	The deferred steps are not well known
Op8	The hook operations are not well identified
Pr1	The object-oriented paradigm is not well mastered
Pr2	The reusability objectives are not well defined
Pr3	The extensibility objectives are not well defined
St3	The programming language is not well mastered

TABLE V. ARGUMENTS RELATED TO THE TM

Id	Issue or Claim	Rea	Reb	All	Dim	Act
1	A final method cannot be overridden by subclasses		7		+ FQ	DI
6	The low cohesion reduces the analysability of			9	- BPQS	ADI
1 1	The low cohesion makes maintenance more tedious	5, 6			- BFPQS	ADIM
1 5	The template method is subject to change	22, 24, 25, 26			- BQS	DI
2 2	The extensibility objectives are not well				- BQS	ADI
2 4	There are too many primitive methods				- Q	DI
2 5	The deferred steps are not well known	22			- BQS	DI
2 9	The hook operations are not well identified	22			- BFQS	DI

The arguments that relate to the most prioritized activities and dimensions produced more remarkable values in the contextual (i.e., quantified) view. As a second step, the work team estimated how much each argument was relevant to the project. The ranking of the arguments generated the concrete quantified views. As a result, the arguments were then contextualized and their weights calculated. Each argument is potentially the root of an argumentation with reasons, rebuttals, and alleviations. Therefore, the weight of an argumentation is the sum of its rank (H, M, or L) and the ranks of its constituting arguments divided by the number of nodes in the argumentation.

C. Identifying critical factors

Weighting activities, dimensions, and arguments generated contextual views that were used to identify critical factors of the project, which correspond to the cells of views that have remarkable values. The cells were prioritized based on their values. A total impact value was computed for each cell by summing the multiplied weights of the arguments it contains. These values were translated into priorities (1 is the highest priority). Our approach suggested reasoning further about the factors that relate to the most prioritized cell in order to nullify or reduce its value. We made the assumption that taking actions to address these most influent factors produces the greatest benefit.

After the critical factors were addressed, their ranking was adjusted. The adjusted rankings provided new priorities. The user iterates these steps (i.e., identifying flaws and taking actions) until satisfied with the values in the concrete views (i.e., specific threshold values attained).

One of the experiments where the SAM framework was applied was an undergraduate course of object-oriented software design at ETS [11]. The project analyzed in this experiment focused on the design and implementation of a software framework that provided the skeleton of a dice game (DGSF). Table VI presents a contextualization of the factors. Table VII presents a view for the DGSF. In addition, a tool-support was used for managing the SDAs, SSMs, and arguments of the case study. The SDAs manager was developed using the Java programming language and Eclipse development platform. The SSMs and arguments manager was developed using a Java-based compiler and a grammar.

V. RELATED WORK

Many organizations maintain SDAs and AK in a database to assist document control, development, and maintenance activities. Much AK and support for designers provided in the literature includes design decision, design rationale, pattern, tactic and quality model [1, 2, 3, 4, 7, 8, 9, 10]). However, most of these models, methods, and tools provide limited views into the AK database [3, 7, 8]. Many approaches have been proposed to support the design process [2, 3, 7], but few [8] support designers to manage and keep track of the AK. We believe our approach can be used to describe SDAs in a manner that may facilitate selecting relevant AK to keep track of selected SDAs as quantified design decisions. The SAM framework can be used to analyze the AK using structured views that relates in a finer-grained manner the artifacts of the problem space to those of the solution space, from organizational goals to specific system artifacts. We believe a multi-dimensional view is a valuable artifact for providing an integrated view of architectural knowledge.

VI. CONCLUSIONS

This paper presented a SAM framework that supports AK management and a multi-dimensional analysis approach to analyzing SDAs such as design patterns and related AK. The proposed AK model describes AK using a set of factors of influence such as SDAs, issues, arguments, activities, and dimensions. Relating these factors enabled the creation of multi-dimensional views that support designers in identifying and addressing critical factors to their projects. The approach was used in industrial and academic contexts. As a proof of concept a prototype tool was developed that students used for analysis. The case studies and controlled experiments produced evidence that the multi-dimensional analysis approach supported by the SAM framework is a valuable step towards handling SDAs and the related AK as an integrated set of factors of influence. The proposed approach may be customized to better support particular SAD processes and system needs. In the near future, it will be supported by an AK management tool. One goal of this work was to contribute to building an AK model of factors linked formally and exploited by algorithms. Finally, five case studies and three controlled experiments have been conducted and will be presented in a forthcoming paper.

TABLE VI. RANKING FACTORS FOR THE DGSF

Ranking of activities for analysis		Ranking of arguments for each iteration			
		Arg.	Iter1	Iter2	Iter3
Architecting	M	1	L	L	L
Designing	H	2	H	H	H
Implementing	M	6	M	L	X
Managing	L	7	L	X	X
Ranking of aspects for analysis		9	H	H	H
Budget	X	11	X	X	X
Functions	M	15	L	L	L
People	M	22	H	L	X
Quality	H	24	X	X	X
Schedule	M	25	H	X	X
		29	X	X	X

TABLE VII. CONCRETE VIEWS OF DGSF ARGUMENTS

Activity	Aspect			
Iteration 1	F	P	Q	S
A	10	11	3	8
D	6	5	1	2
I	13	12	4	7
M	16	15	9	14
Iteration 2	7	6	1	5
A	14	13	16	12
D	11	10	15	9
I	8	4	2	3
M				

REFERENCES

- [1] Gamma, E., Helm, R., Johnson, R., Vlissides, J., "Design Patterns: Elements of Reusable Object-Oriented Software", A.-W., B. (1995).
- [2] Clements, P., Bachmann, F., Bass, L., Garlan, D., Ivers, J., Little, R., Nord, R., Stafford, J., "Documenting Software Architectures – Views and Beyond", Addison Wesley, Boston (2003).
- [3] Bass, L., Clements, P., Kazman, R., "Software Architecture in Practice", Addison Wesley, Boston (2003).
- [4] Standard, I.: ISO/IEC 42010 Systems and Software Engineering – Recommended Practice for Architectural Description of Software-Intensive Systems. ISO/IEC 42010, (2011).
- [5] Carlos, C., Jarred, M., Sanjay, M., Iyad, R., Chris, R., Guillermo, S., Matthew, S., Gerard, V., Steven, W., "Towards an argument interchange format", *Knowl. Eng. Rev.* 21, 4 (Dec. 2006), 293-316.
- [6] Wiegers, K.E., "Standing on Principle," *Journal of the Quality Assurance Institute*, vol. 11, no. 3 (July 1997).
- [7] Kim, S., Kim, D.K., Lu, L., Park, S., "Quality- driven Architecture Development Using Architectural Tactics", *Journal of Systems and Software* 82, 1211- 1231 (2009).
- [8] Ovaska, E., Evesti, A., Henttonen, K., Palviainen, M., Aho, P., "Knowledge Based Quality-driven Architecture Design and Evaluation", *Journal of Info. and Soft. Tech.* 52, 577-601 (2010).
- [9] Shahin, M., Liang, P., Khayyambashi, M.R., "Architectural Design Decision: Existing Models and Tools", In: *WICSA/ECSA 2009*, pp. 293-296. IEEE, Cambridge (2009).
- [10] Parizi, R.M., Ghani, A., "Architectural Knowledge Sharing (AKS) Approaches: a Survey Research", *Journal of Theoretical and Applied Information Technology*, 1224–1235 (2008).
- [11] Adam, S., El-Boussaidi, G., "A multi-dimensional approach for analyzing software artifacts", 25th SEKE, June 27-29, Boston (2013).
- [12] Adam, S., El-Boussaidi, G., Abran, A., "An approach for classifying design artifacts", 27th SEKE, June 6-8, Pittsburgh (2015).

RARep: a Reference Architecture Repository

Tales Prates Correia*, Milena Guessi[†], Lucas Bueno Ruas Oliveira^{*‡} and Elisa Yumi Nakagawa*

*University of São Paulo - USP, São Carlos, Brazil

[†]University of South Brittany - UBS, Vannes, France

[‡]Federal Institute of São Paulo - IFSP, São Carlos, Brazil

Email: tales.correia@usp.br, {milena,oliveira,elisa}@icmc.usp.br

Abstract—Reference architectures are a special type of software architectures that have been proposed for supporting standardization, development, and evolution of software systems of a given domain. As these architectures could contribute for knowledge reuse and increased productivity, several companies have been creating specific reference architectures for their field of expertise. Nonetheless, there is no mechanism that enable recovering, publishing, and sharing existing reference architectures for the public. The main contribution of this paper is to present RARep (Reference Architecture Repository), a web-based reference architecture repository supporting the dissemination of materials related to reference architectures. As a result, this tool can facilitate the access to information on reference architectures and, hence, promote the sharing of architectural knowledge contained in such architectures.

Keywords—Software architecture, reference architecture, web-based tool.

I. INTRODUCTION

Software architectures have played a central role in the development of successful software systems over the past 20 years [15, 25]. Garlan and Perry [10] state that software architectures can have a positive impact in many aspects of software development, such as understanding, reuse, evolution, analysis, and management. Thereby, software architectures also play an important factor for guaranteeing software systems quality [6], such as maintainability, dependability, and interoperability [26].

As a particular type of software architectures, reference architectures have stood out as a structure that provides a characterization for software systems functionalities of a given domain [6, 17]. In other words, reference architectures encompass knowledge about a given domain, which can offer important information on how to build, develop, and maintain systems for that specific domain. In this scenario, the availability of reference architectures becomes an important concern since it impacts the dissemination and reuse of knowledge contained in these architectures as well as the productivity of software development processes. Considering its relevance, a variety of reference architectures for different domains can be found, such as the ones presented at [2, 5, 8, 11, 24]. Reference architectures has also been explored for the domain of embedded systems, such as automotive [4], ambient assisted living [1, 20, 23], and robotics [13].

In this perspective, the main motivation for developing an open repository is the fact that a mechanism to catalog, promote, and share reference architectures is still missing.

Moreover, even though many reference architectures are published in articles, technical reports, or thesis, they are not easily available to the interested public. Thereby, we have designed a tool that supports the software architecture community to have easier access to information related to reference architectures, creating an open knowledge repository for software systems developers. Thereby, this repository can be used as an index for several types of material related to reference architectures aiming to facilitate the recovery and selection of reference architectures for the software design.

This paper is organized as follows. Section II introduces references architectures and details relevant works motivating the development of our repository. Section III discusses the requirements of this tool as well as its conceptual design. Section IV details how this repository is implemented and its main features besides an illustrative example of its use. Section V discusses some perspectives for extending this repository in future research. Finally, Section VI presents our final remarks.

II. BACKGROUND

Decisions made at the architectural level can directly enable, facilitate, or interfere in the achievement of business goals, as well as functional and quality requirements [17]. Several terms are used to designate software architectures in different abstraction levels. In particular, the term reference model is frequently misused as a synonym for reference architecture. A reference model designates a structure that promotes the understanding on a given domain by sharing a common vocabulary and the parts and its interrelationships without considering implementation details [6]. The OASIS Reference Model for Service Oriented Architectures¹ is an example of reference model. On the other hand, reference architectures are less abstract than reference models as they provide concrete guidelines for the design of software architectures, such as architectural styles, best practices for software development, and software elements supporting the development of systems [18].

In this scenario, a reference architecture can combine reference models and architecture patterns and, hence, support the creation of several concrete software architectures that pertain to a given domain (Figure 1). Aiming to systematize

¹OASIS, <https://www.oasis-open.org/committees/download.php/19679/soa-rm-cs.pdf>

the creation of reference architectures, several approaches have been proposed. For example, Muller (2008) [16] provides guidelines for establishing reference architectures. Angelov et al. (2012) [3] propose a classification framework for reference architectures that aims at promoting the analysis and design of reference architectures by assessing their adherence to this framework. Nakagawa et al. (2014) [19] present a process to establish, represent, and evaluate reference architectures that is focused on improving separation of concerns in such architectures.

Furthermore, we observe the proposition of a reference model for reference architectures, called RAModel [17], which aims at promoting a better understanding about the content of reference architectures in terms of their main elements and relationships with each other as well as the main tasks for establishing, using, and evolving such architectures. According to the RAModel, a reference architecture contains information about the domain, application, and infrastructure, as well as crosscutting information that is related to several of these dimensions.

The description of reference architectures is fundamental for their adoption and effective use during the software development process. In particular, the standard ISO/IEC/IEEE 42010:2011 specifies the main elements of an architectural description, the relationship with each other, and their organization [14]. The most important elements are: model type, architectural view, viewpoint and correspondence. Since reference architectures are more generic than concrete software architectures, current practices for describing software architectures must be tailored for reference architectures [12]. These reference architectures are often described at a higher abstraction level, have no clear stakeholders, involve more architectural qualities, and have a larger scope.

Aiming at disseminating reference architectures to the interested public, an open knowledge repository is needed. This repository can help to publish, share, and find reference architectures that have been created for a given domain of expertise or that follow the same design principles. The design and implementation of such a repository can be inspired in currently available tools, such as one for robotics services semantic search tool, called RoboSeT², and another for software patterns, called Portland Pattern Repository³, for devising a set of relevant features for a novel reference architectures repository.

III. REPOSITORY FOR REFERENCE ARCHITECTURES

Considering the relevance of reference architectures for promoting best practices in software systems development, we propose a software tool supporting the dissemination of reference architectures, hereinafter referred to as RaRep. This tool contains particular features allowing the interaction among users of the repository. For instance, users of this tool can post comments and up-vote reference architectures, which enables

to establish a bidirectional communication channel between creators of a reference architecture and its public. Furthermore, the repository can be used as an index of reference architectures that is directly managed by the creators of reference architectures. As a consequence, we are able to provide an up-to-date catalog of reference architectures for several domains that make available their architectures.

Taking into account the RAModel and the standard ISO/IEC/IEEE 42010:2011, we specify which reference architectures concepts can be used in RaRep for documenting reference architectures. This specification is important since the repository's main goal is to act like a broker of reference architectures, i.e., enabling creators to share their own reference architectures or look for one or more that they could use in their project. In particular, we designed UML [22] models about the main concepts and functionality of this tool. Figure 2 shows the conceptual model for this repository, which identifies the main elements related to the creation and use of reference architectures. This representation simplifies the way users can post, select, and navigate reference architectures in our repository. Figure 3 shows the UML use case diagram for RARep, in which the main actors and actions are defined. In particular, we devise three types of actors in our tool (i.e., common users, registered users, and admin users) with different levels of permissions. For instance, only registered users can publish, comment, and post news about registered reference architectures.

This representation, however, does not preview more specific reference architecture elements such as business rules, constraints, risks, goals and needs. These elements are more related to the knowledge itself encompassed in a reference architecture description. Other elements that can be categorized as the core of architectural knowledge management is the architecture rationale, which represents the decisions made over the project and the alternatives of that decisions [17]. In another context, the main goal of architectural knowledge management is to prevent knowledge vaporization in software architectures. To do so, architecture rationale for significant architectural decisions made in the software architecture should also be included in the architecture description. Significant architectural decisions could be for example the selection of a particular concern or viewpoint, the definition of the most adequate abstraction level for a particular viewpoint, or the reason for adopting a particular design pattern. Furthermore, several aspects of an architectural decision can be relevant, such as their implications to the software architecture design, constraints and rules imposed by them, and also the reasoning that lead to them [7]. Therefore, architectural decisions play an important role for education, reuse, and evolution of software architectures as they are used for sharing expertise and best practices. In the context of reference architectures, significant architectural decisions encompass guidelines for deriving the reference architecture into concrete software architectures besides documenting the architectural knowledge of concrete software architectures of a given domain. Hence, reference architectures certainly need to address architectural decisions

²RoboSeT, <http://www.labes.icmc.usp.br:8595/RegistroServicoWeb/>

³Portland Pattern Repository, <http://www.patternrepository.com>

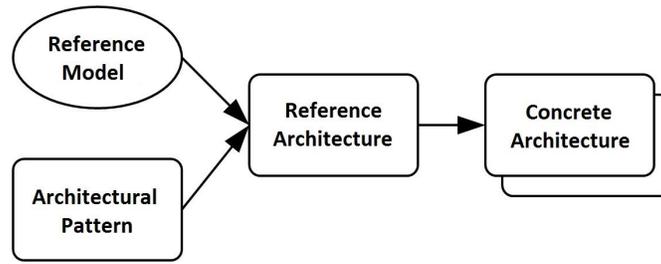


Fig. 1. Relationship among reference model, architectural pattern, reference architecture, and concrete architecture [6]

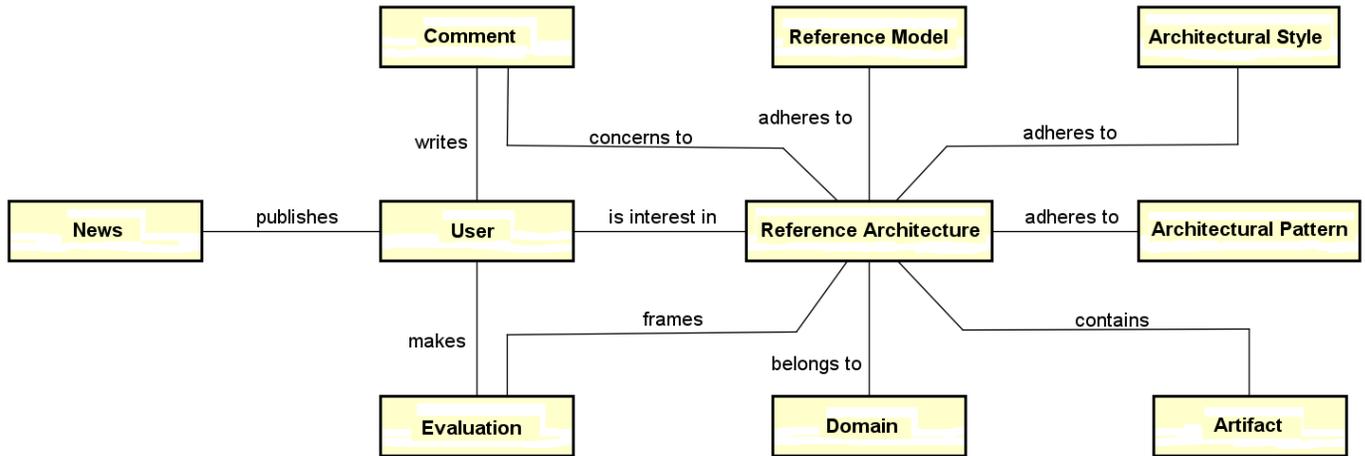


Fig. 2. Conceptual model of the reference architecture repository

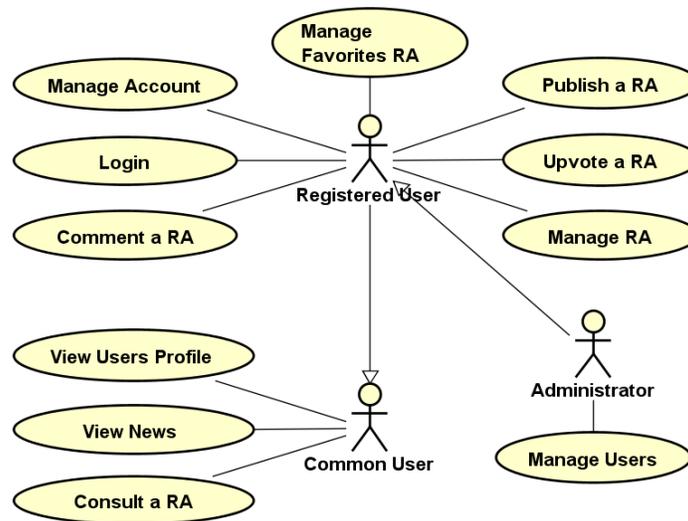


Fig. 3. Use case diagram of the reference architecture repository

in their architectural description.

IV. DEVELOPMENT

This section details how this repository is developed. In particular, RARep⁴ is implemented using current well known

frameworks and technologies to the development of web systems, such as Java, MySQL⁵, Hibernate⁶, and Bootstrap⁷.

A. Architectural Project

Using the conceptual model and use case diagram previously presented as baseline, we developed a software tool

⁴RARep, <http://www.labes.icmc.usp.br:9083/RARep/index.jsp>

⁵MySQL, <https://www.mysql.com/>

⁶Hibernate, <http://hibernate.org/>

⁷Bootstrap, <http://getbootstrap.com/>

supporting reference architectures dissemination. These concepts were important for the development of a tool prototype that was used in the identification of additional features and refinement of these models. In particular, this tool standardizes the creation, presentation, and organization of reference architectures. Moreover, some of the features supported in RaRep are inspired in discussion forums and similar repositories, such as an up-vote feature and different search mechanisms. The registered user inherits the common user features and the administrator inherits the registered user features. The features related to the common users are: (i) view news, where the user can visualize news present in the repository; and (ii) consult reference architecture, where the user can see all details of a reference architecture available in the repository, such as the ones related to a reference architecture.

The features related to the registered users are: (i) login, which the user can authenticate in the repository; (ii) manage account: where the user can update his personal information; (iii) manage favorites reference architectures, which the user can add, remove or list his favorites reference architectures; (iv) up-vote a reference architecture, which the use can up-vote a reference architecture present in the system; (v) comment a reference architecture, which the user can post a comment in a reference architecture in the repository; and (vi) manage reference architectures, which the user can add, remove, and update its reference architectures in the repository. Administrators can also manage users, adding, removing, and updating every information in the repository.

B. Implementation

As mentioned before, software prototyping was used to mitigate uncertainties in the requirements. During the development process, the first part implemented was the front-end of the website. At this stage, all pages were developed as well as the transitions between them. The next step was the development of the back-end of the repository. At this step, we implemented all Hibernate XML for data persistence in the database. The conceptual model was used for mapping the classes of the repository. Then, we developed the update, insert, select and remove methods of these classes. After the development of the front-end and back-end, the next step was to integrate these two parts. The latest repository deployment has a total of 14 classes, around 5 kloc, and 22 hibernate XML files.

Similar software engineering tools helped us to come up with the features currently available in the reference architecture repository. The main activities of this repository are post and search for reference architectures. Aside from that, there are several features supporting the interaction among users of this repository, such as: (i) posting comments about a particular reference architecture; (ii) listing a particular reference architecture in its favorite list; (iii) evaluating a reference architecture by means of an up-vote system; and (iv) posting news about their own reference architectures. These features are available in a navigation bar at the top of the page. Apart from the search feature, the other features are only available to registered users of the system. We made

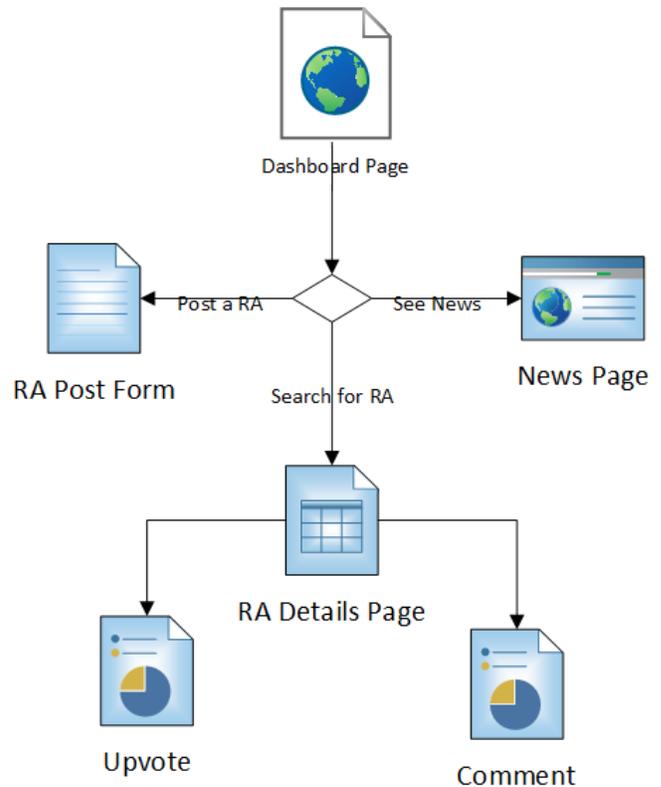


Fig. 4. Navigation flowchart for the repository

this decision because the main goal of this repository is to disseminate reference architectures to whoever access the repository. Figure 4 shows a navigation flowchart describing the different actions that an user can perform in this repository.

C. Illustrative Example

In this subsection we present an illustrative example using our repository to register a reference architecture. RefSORS (Reference Architecture for Service-Oriented Robotic Systems)[21] is a reference architecture for developing indoor, grounded mobile Service-Oriented Robotic Systems (SORS), i.e., robotic systems designed according to Service-Oriented Architecture (SOA). In order to post RefSORS, the first step is to login into the repository. In case the user is not registered into the repository, then the user has to fill a form with his account info such as full name, login, email, institutions he is affiliated and password. The next step is select the *post a reference architecture* feature at the navigation bar. At this point, the form presented in the Fig. 5 will be available to the user to complete the information of his reference architecture. For the RefSORS, the form can be filled in as:

Title: Reference Architecture for Service-Oriented Robotic Systems - RefSORS;

Authors: Lucas Bueno R. Oliveira, Elisa Yumi Nakagawa, and Flavio Oquendo;

Institutions: ICMC/USP (Brazil), UBS (France);

Date: Oct 10, 2014;

Domains: Robotic Systems, Embedded Systems;

Architectural styles: SOA;

Reference Models: OASIS Reference Model

Architectural Patterns: Layered Architecture;

Related Documents: SOA-RA technical standard, OASIS reference architecture, ArchSORS process and SoaML;

After posting a reference architecture in the repository, all registered users can discuss the architecture by posting comments. In addition, users can also upvote, favorite, and search for this architecture.

V. FUTURE ACTIVITIES

This work was motivated by the need of a supporting tool that promotes dissemination of reference architectures. This task is not simple given the diversity and amount of knowledge encompassed in reference architectures. As future activities, we plan to:

- **Improve reference architectures documentation:** increasing the amount of details and artifacts present in the representation of reference architectures without jeopardizing its clearance and easiness to identify its element. These new elements, for example, could be related to architecture decisions. As a consequence, instantiating the reference architecture into concrete software architectures could be resumed to the selection of architecture decisions.
- **Develop additional features:** Currently, our repository supports managing several artifacts related to reference architectures documentation and search for particular information regarding their design. However, we intend to develop additional features that could enhance users' experience with our tool. For instance, we intend to support the communication among different tools with RARep such as Research Gate⁸, so users will be able to link account to RARep. We expect to identify other relevant features in a case study regarding the usability of this tool.
- **Treat variability of reference architectures:** The variability in reference architectures concerns the ability of a software artifact built from such architectures to be adapted for a specific context in a preplanned manner [9]. In this sense, documenting variability to architecture decisions and the reference architecture itself would help to create an even larger knowledge repository as all alternatives, dependencies, and options would be registered in the reference architecture.

VI. CONCLUSION

Reference architectures are key to knowledge reuse in software systems as they promote best practices. Despite their relevance, a mechanism supporting their open distribution to the public was still missing. The main contribution of this paper is to present the design and implementation of a web-based tool that supports the creation of a catalog of reference

architectures. This web-based tool takes into account a reference model for reference architectures as well as best practices for software architecture description, including architectural viewpoints, styles, and concerns that have been considered in their design.

As future work, we intend to extend this web-based tools with additional features. As a consequence, we expect that this tool can contribute for further disseminating and facilitating the access of information related to reference architectures and, hence, promoting knowledge reuse.

ACKNOWLEDGMENT

This work is supported by the Brazilian funding agency FAPESP, grants 2014/02244-7, 2014/25341-8, and 2012/24290-5.

REFERENCES

- [1] C. H. Alliance. *Continua Health Alliance*. On-line. Available at: <http://www.continuaalliance.org/> (02/2016). 2013.
- [2] S. Angelov, P. Grefen, and D. Greefhorst. "A Classification of Software Reference Architectures: Analyzing Their Success and Effectiveness". In: *IEEE/IFIP Conference on Software Architecture (WICSA'09)* (2009), pp. 141–150.
- [3] S. Angelov, P. Grefen, and D. Greefhorst. "A Framework for Analysis and Design of Software Reference Architectures". In: *Information and Software Technology* vol. 54 (2012), pp. 417–431.
- [4] AUTOSAR. *AUTOSAR (AUTomotive Open System ARchitecture)*. On-line. Available at: <http://www.autosar.org/> (Accessed 02/2016). 2013.
- [5] P. Avgeriou, S. Retails, and M. Skordalakis. "An Architecture for Open Learning Management Systems." In: *Panhellenic Conference on Informatics (PCI'2003)* (2003), pp. 183–200.
- [6] L. Bass, P. Clements, and R. Kazman. *Software Architecture in Practise*. Addison-Wesley, 2012.
- [7] J. Bosch. "Software Architecture: the Next Step". In: *First European Workshop: Software Architecture (EWSA'04)* (2004), pp. 194–199.
- [8] N. S. Eickelmann and D. J. Richardson. "An Evaluation of Software Test Environment Architectures". In: *International Conference on Software Engineering (ICSE'96)* (1996), pp. 353–364.
- [9] M. Galster et al. "Variability in Software Architecture: Current Practice and Challenges". In: *SIGSOFT Software Engineering Notes* vol. 36 (2011), pp. 30–32.
- [10] D. Garlan and D. Perry. "Introduction to the Special Issue on Software Architecture". In: *IEEE Transactions on Software Engineering* (1995), pp. 269–274.
- [11] A. Grosskurth and M. W. Godfrey. "A Reference Architecture for Web Browsers". In: *IEEE International Conference on Software Maintenance (ICSM'05)* (2005), pp. 661–664.
- [12] M. Guessi, L. B. R. Oliveira, and E. Y. Nakagawa. "Representation of Reference Architectures and Reference Models: A Systematic Review". In: *28th Int. Conference on Software Engineering and Knowledge Engineering (SEKE'11)* (2011), pp. 1–4.
- [13] M. Hagele. *Project RoSta - Robot Standarts and reference architectures*. On-line. Available at: <http://www.robot-standarts.org/> (Accessed 02/2016). 2013.
- [14] ISO/IEC/IEEE. *ISO/IEC/IEEE 42010:2010 International Standard for Systems and Software Engineering – Architectural description*. 2011.
- [15] P. Kruchten, H. Obbink, and J. Stafford. "The past, present, and future of software architecture". In: *IEEE Software* vol. 23, n^o 2 (2006), pp. 22–30.
- [16] G. Muller. *A Reference Architecture Primer*. On-line. Available at: <http://www.gaudisite.nl/ReferenceArchitecturePrimerPaper.pdf> (Accessed 02/2016). 2008.
- [17] E. Y. Nakagawa, F. Oquendo, and M. Becker. "RAModel: A Reference Model of Reference Architectures". In: *IEEE/IFIP Conf. on Software Architecture & Eur. Conf. on Software Architecture (WICSA/ECSA'2012)* (2012), pp. 297–301.

⁸Research Gate, <https://www.researchgate.net/>

Post reference architecture

Title	Reference Architecture for Service-Oriented Robotic Systems - RefSORS
Authors	Lucas Bueno R. de Oliveira Elisa Yumi Nakagawa Flavio Oquendo
Institutions	ICMC/USP UBS France
Description	<p>RefSORS (Reference Architecture for Service-Oriented Robotic Systems) is a reference architecture for developing indoor, grounded mobile Service-Oriented Robotic Systems (SORS), i.e., robotic systems designed according to Service-Oriented Architecture (SOA). It is described using both high-level, general representation and semi-formal languages SoaML (Service-oriented architecture Modeling Language) and UML (Unified Modeling Language). RefSORS is represented in several levels of abstraction and encompass the following</p>
Date	17 March 2016
Domains	Robotic Systems Embedded Systems
Architecture Styles	SOA
Reference Models	OASIS Reference Model
Related Patterns	Layered Architecture
External Documents	SOA-RA technical standart OASIS Reference Architecture ArchSORS process SoaML
	<input type="button" value="Post"/>

Fig. 5. Reference architecture post form

- [18] E. Y. Nakagawa, F. Oquendo, and J. C. Maldonado. "Software Architecture: Principles, Techniques, and Tools". In: ed. by M. Oussalah. John Wiley & Sons, 2015. Chap. Reference Architectures, pp. 101–122.
- [19] E. Y. Nakagawa et al. "Consolidating a Process for the Design, Representation, and Evaluation of Reference Architectures". In: *Working IEEE/IFIP Conference on Software Architecture (WICSA'14)* (2014), pp. 1–10.
- [20] OASIS. *Reference Architecture Foundation for Service Oriented Architecture Version 1.0*. On-line. Available at: <http://docs.oasis-open.org/soa-rm/soa-ra/v1.0/cs01/soa-ra-v1.0-cs01.html> (Accessed 02/2016). 2013.
- [21] L. B. R. Oliveira et al. "Towards a Process to Design Architectures of Service-Oriented Robotic Systems". In: *European Conference on Software Architecture (ECSA'14)* v. 8627 (2014), pp. 218–225.
- [22] OMG. *Unified Modeling Language v. 2.4.1*. [On-line]. Available at: <http://www.omg.org/spec/UML/2.4.1/> (Accessed 02/2016). 2011.
- [23] U. Project. *The UniversAAL Reference Architecture*. On-line. Available at: <http://www.universaal.org/> (Accessed 02/2016). 2013.
- [24] K. Sandkuhl and B. Messer. "Towards Reference Architectures for Distributed Groupware Applications". In: *Euromicro Workshop on Parallel and Distributed Processing* (2000), pp. 135–141.
- [25] M. Shaw and P. Clements. "The Golden Age of Software Architecture". In: *IEEE Software* 23.nº 2 (2006), pp. 31–39.
- [26] A. I. Wasserman. "Towards a Discipline of Software Engineering". In: *IEEE Software* vol. 13.no. 6 (1996), pp. 22–31.

A Multi-Agent Architecture for Quantified Fruits: Design and Experience

Jean-Pierre Briot^{*†}

Nathalia Moraes do Nascimento[†]

Carlos José Pereira de Lucena[†]

(^{*})Sorbonne Universités, UPMC Univ Paris 06, CNRS
Laboratoire d'Informatique de Paris 6
Paris, France
jean-pierre.briot@lip6.fr

([†])Laboratório de Engenharia de Software
Departamento de Informática, PUC-Rio
Rio de Janeiro, Brazil
nnascimento,lucena@inf.puc-rio.br

Abstract - *The concept of Quantified Self is about connected objects self-monitoring their human owner (e.g., a watch measuring heart rate, etc.). A natural transposition is in self-monitoring arbitrary things, therefore named Quantified Things. In this paper, we present the case of self-monitoring agricultural products. We discuss the rationales for the design of a Quantified Fruit multi-agent architecture for self-monitoring and self-prediction of the maturation of fruits. The architecture includes 6 different types of agents, the 2 more specific ones being respectively, the self-controller equipped with various sensors and the self-prediction module. Our current implementation uses an Arduino microcontroller board with 5 sensors (measuring respectively: temperature, light, humidity, hydrogen and methane). The prediction module uses a neural network. We have implemented the architecture and have conducted various experiments, storing bananas in diverse settings: room, refrigerator, in a box, with other fruits, etc. The paper discusses the architecture, its current implementation, experiments and current results. Future issues (scalability, collaborative prediction, etc.) are also addressed.*

Keywords – **Quantified Self; Quantified Things; Internet of Things; microcontroller; software; architecture; design; implementation; agent; multi-agent system; monitoring; prediction; machine learning; neural network; fruit; banana; maturation; logistics.**

1. Introduction

As Swan [1] suggests, the concept of *Quantified Self* (QS) represents the capacity for connected objects to self-measure and self-monitor their human owner. Examples are connected watches or phones that measure heart rate, pressure, exercising habits, etc. Capacities for analysis, patterns detection and prediction (using statistical analysis and machine learning techniques) may be included in order to infer personalized monitoring and diagnostic, e.g., for health monitoring.

In this work, we investigate the adaptation of this idea to arbitrary things, therefore named *Quantified Things* [2]. Some early examples are Quantified Cars [3], using the large electronic monitoring and control facilities of a car

to monitor, diagnose and control various features of a car. Swan suggests the use of “QS car chips” to collect cars automotive data and store these informations in a cloud database. By using a mobile application, users could have access to their car’s information, such as maintenance records, suggested and scheduled maintenance, and take more accurate action as a result.

We have decided to address the case of agriculture food products which, to our knowledge seems a domain yet little explored. In particular, the lifecycle of fruits has an important impact on its economy. An important issue is indeed to minimize the loss of fruits too mature to be consumed and at the same time to minimize the risk of shortage of products for the consumers. This is specially true in the case of bananas, a fruit having a relatively short ripening period, and very much depending on various conditions (temperature, humidity, light, aeration) [4]. Therefore, important decisions must be taken at various steps of the lifecycle: when to best harvest the fruits, depending of the expected travel (type and duration) to the consumer, how to best transport them, how to store them, at a large scale in a storage or a grocery store, down to the consumer house, etc.

In order to explore these issues, we have designed a prototype multi-agent architecture for Quantified Fruits. Its objective is self-monitoring and self-prediction of fruit maturation. We have tested the architecture in the case of bananas and have evaluated it as a proof of concept. The proposed architecture includes various types of agents, implemented in the JADE framework: an Arduino programmable microcontroller board with various sensors (temperature, light, humidity, hydrogen, methane), a user interface and a neural network-based prediction module. We have selected these sensors based on some works [5, 4] that investigate various factors that interfere on fruit’s perishability.

2. Related Work

Some researchers have proposed the use of sensors in the agriculture supply chain. Most of them investigate how technology can be used to improve farming practices, such as collecting information about the land conditions and climate variability and helping farmers to avoid inappropriate farming conditions [6, 7]. However, even though the percentage of agriculture food products losses after picking

step is extremely high, there have been very few investigations into what are the satisfactory conditions to prolong perishable shelf life in the other supply chain steps, such as distribution. Lang et al. [8], for example, suggest the development of an intelligent container for monitoring temperature parameter during food transportation. Nonetheless, they do not present experimental results. In addition, their model takes only temperature into account for monitoring the quality of the product.

Examples of software architectures for IoT are the Fed-Net service-based framework [9] and the multi-agent-based middleware ACOSO [10]. As we will see, our architecture is more light weight and more specific to our objective.

3. Application Scenario

3.1. Setting

In our first scenario, we consider a user having at home a set of bananas that he bought and he would like to know what are the best ways to store bananas and how to predict their maturation. We have selected the case of bananas, because: 1) the interaction between the lifecycle of tropical fruits and the logistics for maturing, transporting and storing them is critical and with an economic impact; 2) they have a relatively short lifespan, therefore we can conduct shorter experiments. Different ways of storing bananas may be considered and tested: in a room, in a refrigerator, in a box, alone or with other fruits, etc. A self-monitoring device (described below) is associated to a banana (located on its side) in order to provide the self-monitoring capacity.

3.2. Requirements

Some of the requirements for our application are:

- a software architecture to autonomously provide self measurement and prediction of fruit maturation. It should provide some pro-activeness capacities, such as to self assess its prediction accuracy and if necessary revise its prediction model.
- the architecture must be decentralized, interoperable and light weight, in order to work with different types of microcontrollers, resources and constraint settings (local computing, cloud, network protocol, etc.).
- the architecture should be easy to deploy (with some auto-deployment and configuration ability) and evolvable at design time as well as run time (discovery of new quantified things devices and modules, specially in the context of collaborative applications, see § 6.4).
- a user interface running on a smartphone (or tablet) for the user to control experiments and input some data.

4. Proposed Architecture

4.1. Conceptual Architecture

In order to offer a decentralized and collaborative architecture with such requirements (autonomy, pro-activity, decentralization, inter-operability, easiness to deploy, and furthermore ability to seamlessly incorporate future evolutions, such as collaborative Self Things, see § 6.4), a natural choice was to use a multi-agent approach (see for instance [11] for a more detailed argumentation). Therefore, the various autonomous modules (monitoring, learning module, data base, user interface, etc.) are encapsulated into various interacting agents. The conceptual architecture proposed is shown at Figure 1. It includes the following components:

- QuantifiedFruitAgent, which encapsulates the microcontroller and its sensors used for self-monitoring.
- PredictionAgent, which encapsulates the machine learning module to make the prediction.
- TrainingAgent, which encapsulates the training algorithm which will set up PredictionAgent parameters.
- DataBaseAgent, encapsulating the various data of the experiments, notably the training set.
- UserInterfaceAgent, encapsulating the interface with the user.
- PlugAndPlayAgent, responsible for discovering a new QuantifiedFruit, creating a new QuantifiedFruitAgent, configuring and connecting it to the architecture components.

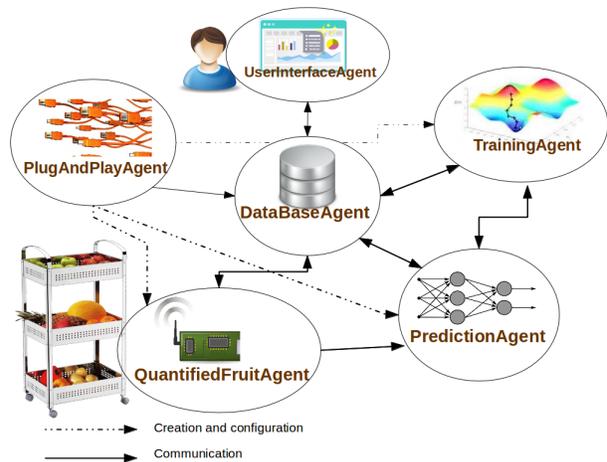


Figure 1: Conceptual architecture.

PlugAndPlayAgent creates a QuantifiedFruitAgent for each Arduino board that is connected to the system and associates it with a specific PredictionAgent.

DatabaseAgent is shared among all agents, allowing them to share information, such as the self-tracking and prediction data. It could be noted that the database does not need to be an agent, but encapsulating it into an agent (agentifying it) brings uniformity to the architecture and the communication between its components, as well a potential of flexibility for the future.

4.2. Current Implementation and Configuration/Deployment

Current implementation and configuration/deployment of the architecture is as follows. Each agent is implemented as a JADE agent, using the JADE multi-agent infrastructure [12]. JADE provides the support for the interoperability and the distribution of the agents.

- QuantifiedFruitAgent encapsulates an Arduino microcontroller [13] and its 5 sensors, respectively measuring: methane, hydrogen, temperature, humidity, and light. It runs on a Java server. There is one QuantifiedFruitAgent for each Arduino board connected to the system.
- PredictionAgent encapsulates an artificial neural network (ANN) used for prediction. It runs on a Java server. There is one PredictionAgent, shared by the different QuantifiedFruitAgent agents.
- TrainingAgent encapsulates both backpropagation and prediction error minimization algorithms for training the neural network [14]. It runs on a Java server.
- DataBaseAgent encapsulates a data base containing the various data of the experiments. It is currently implemented as a simple table. There is one DataBaseAgent shared by all agents.
- UserInterfaceAgent encapsulates the user-interface running on a smartphone or a tablet, implemented in Java. It runs on a smartphone.
- PlugAndPlayAgent is using a discovery protocol similar to Jini Lookup Discovery Service [15]. It runs on a Java server.

In current configuration, there is only one PredictionAgent and one ConfigurationAgent. This means that there is no heterogeneity and all fruits tested are considered to be of the same type. A larger experiment may introduce different types of fruits and associated PredictionAgent agents.

Note that current configuration of the architecture is obviously not scalable, but its configuration can be adapted as will be discussed in § 6.3. More generally speaking, depending on the availability of resources on the underlying

computing and communication architecture, we may decide to allocate different agents on different spaces (local, global/shared, hierarchical...) and also make various decisions on what agents should be shared or replicated.

Prediction Agent

We have decided to use an artificial neural network (ANN) architecture for the prediction module. The reason is as following: ANNs are well known architectures and they have proven their efficiency and moreover versatility. As opposed to linear or polynomial regression modules where one has to *a priori* select a model (linear, quadratic, cubic, including product of features, etc.), the model of a neural network is generic enough although some configuration has to be decided (e.g. the number of hidden layers, the number of units of the hidden layer(s)).

The neural network includes an input layer with 5 units (corresponding to the 5 parameters produced by the 5 sensors), one hidden layer with 4 units and an output layer with one unit (corresponding to the number of days predicted).

The neural network is implemented in Java. We have also implemented a second version in the Octave/Matlab numerical computation programming language, in order to exploit vectorization of data computation and to conduct further analyses (see § 5.4).

Training Agent

Our current method for training the neural network (adjusting the weights of the neuron connexions in order to minimize the error (differences) between predicted and target values) is quite standard: 1) using backpropagation algorithm (to compute the gradients) [14]; 2) combined with an algorithm to minimize the cost function (prediction error) – we have tried out batch gradient descent well as generic optimization algorithms (from off-the-shelf libraries).

Note that, in addition to traditional off-line learning approach, we also experimented with a (simplified) incremental learning approach, where PredictionAgent proactively self-assesses its prediction accuracy and if necessary requests TrainingAgent to incrementally update its prediction model by launching a new learning phase on the new example(s) (in a similar way to on-line learning).

User Interface Agent

It runs on a Java mobile application in order to allow mobile users to monitor fruit storage. This application lists all fruit storages that are connected to the system. After the mobile user selects one of these fruit storage, UserInterfaceAgent will retrieve from the database all the respective information – monitoring data and prediction – and show it on the interface. In addition, this application has an input field for the user to enter the effective (observed) fruit lifespan. UserInterfaceAgent will then communicate this new experimental data to DatabaseAgent.

5. Experiments and Evaluation

5.1. Experimental Setting

The user will try various ways for storing a banana, taking four condition parameters into account: (i) dark (i.e. in a closed or open box); (ii) room (i.e. the box being stored in a fridge or at room temperature); (iii) rotten fruit (i.e. in a box alone or putting together with a rotten); and (iv) ripe fruit (i.e. putting together with a ripe fruit).

Below, we detail four of the possible settings for storing a banana (summarized at Table 1 and depicted at Figure 2):

- (a) In an open box, at room temperature, alone;
- (b) In an open box, together with a rotten fruit;
- (c) In the fridge, with a ripe fruit;
- (d) In a closed box, at room temperature, with a rotten fruit.

Table 1: Configuration of experiments at Figure 2.

Experiment	Box		Room		Rotten Fruit		Ripe Fruit	
	<i>open</i>	<i>close</i>	<i>room</i>	<i>fridge</i>	<i>yes</i>	<i>no</i>	<i>yes</i>	<i>no</i>
(a)	X		X			X		X
(b)	X		X		X			X
(c)	X			X		X	X	
(d)		X	X		X			X

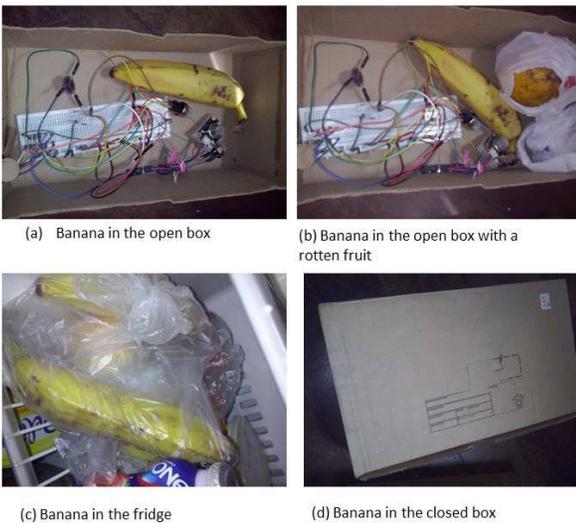


Figure 2: Examples of scenarios.

For each setting, a user creates a new experiment on his smartphone user interface. Then, he triggers the measure-

ment of the parameters (light, temperature, methane, hydrogen and humidity), which will be recorded in the database. He then later checks (usually every day) the maturation of the fruit and reports on the interface when the starting maturation occurs. This process was essential to elaborate an initial database to improve system's predictions. In practice, we have conducted several experiments in parallel, putting a dozen of bananas in different settings and monitoring in parallel their respective maturation process.

Following standard methodology in machine learning, we have partitioned our dataset into a training set and a testing set. (We also have used a cross validation subset for detailed analysis, see § 5.4).

5.2. Training Set

Table 2 shows a subset of the training set used, which represents the data collected from the experiments illustrated at Figure 2. At the beginning of each experiment, QuantifiedFruitAgent collects the measured values: temperature (abbreviated Temp.), which is registered in Celsius (C), relative humidity (RH), hydrogen gas (Hyd.), methane gas (Met.), and luminosity (Lum.). Values of gas sensors are recorded according to the sensor output value (V.). At the end of each experiment, the user reports the "actual" fruit lifespan (this information is subjective since in current experiments naked-eye observation determines it).

Table 2: Subset of the training set.

Temp	RH	Hyd	Met	Lum	Lifespan
27.62	70.22	2	184.0	15.0	14
28.02	72.53	8	275.0	10.0	5
27.81	72.75	3.0	258.0	3.0	10

5.3. Test Set

Table 3 shows results for a subset of the test set. This example, which was performed outside the fridge and in an open box, shows a good prediction. The system predicted thirteen days, and the user reported that the banana spoiled in approximately twelve days.

Table 3: Subset of the test set.

Temp	RH	Hyd	Met	Lum	Lifespan	
					<i>Observed</i>	<i>Predicted</i>
28.21	70.24	3.0	183.0	16.0	12	13

5.4. Results and Discussion

We believe that these first experiments are promising, the prediction module showing good prediction accuracy. Obviously, we need to conduct more experiments with different settings to collect more data.

We have conducted some analysis of our prediction module. Figure 3 shows the validation curve, which compares the evolutions of the prediction error for the training set (we will name it *training error*, depicted in a blue solid line) and of the prediction error for the cross validation set (*cross validation error*, depicted in a green dashed line) for various (increasing) values of λ (the regularization parameter used to control overfitness). The figure shows that 0.01 is a good value for λ as cross validation error is minimal. For a smaller value, there is some variance (overfitness) because the training error is almost null and the cross validation error is significant, showing the poor generalization of the model. For a larger value, the cross validation error is increasing (note that the training error is also increasing), showing an increasing bias (underfitness).

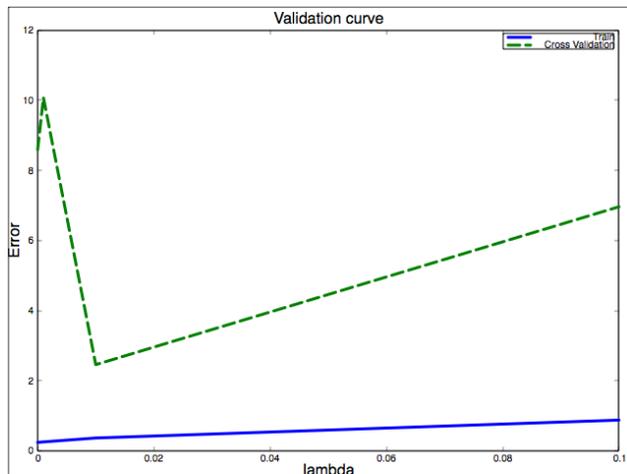


Figure 3: Validation curve.

Figure 4 shows the learning curve, i.e., the evolution of prediction error depending on the size of the training set. The figure shows that the training error is almost null and that the cross validation error stays low, confirming that the model has low bias and low variance. These preliminary analyses are encouraging. We are conducting more experiments in order to collect more data in order to further improve the model.

6. Open Challenges

6.1. Learning Algorithms

The preliminary tests that we have conducted show that the neural network has a good prediction accuracy. But we need to conduct more experiments in order to construct a

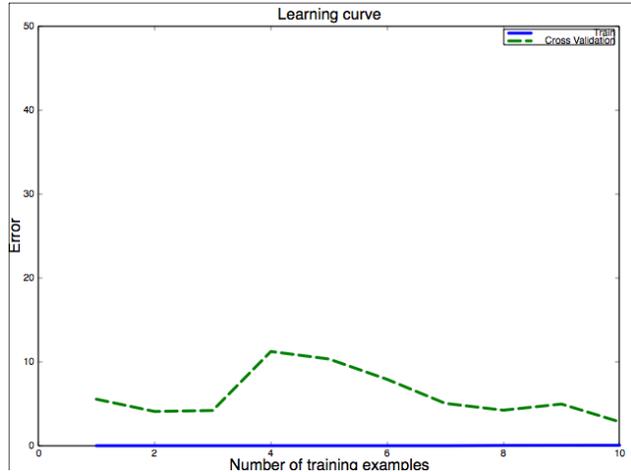


Figure 4: Learning curve.

sufficient and representative data set. We will also continue to conduct analysis of the behavior of the neural network. Note that our architecture is generic and we may consider other types of PredictionAgent, with alternative prediction modules and learning algorithms.

6.2. Genericity

Our architecture is actually generic and may be used for other purposes. Current implementation served as a proof of concept of the architecture. But current implementation provides a unique type of prediction and configuration strategy. It is actually easy to introduce heterogeneity and various kinds of quantified things, prediction and configuration strategies (and associated agents).

Another dimension of genericity is to use the neural network as a controller (and not for prediction). Actually, we have instantiated a preliminary version of the architecture [2] which since then has been completely redesigned, and tested it on a scenario of simulated traffic management, where controllers control semaphores at the crossing of roads. The behavior of the semaphores is evolved using evolutionary algorithms, inspired by evolvable architectures for robotics. This work has been described in [16].

Last, we are planning to reuse and experiment our architecture on other types of applications, such as for instance distributed self monitoring of air pollution in a city through mobile users (e.g., on bicycles and carrying the sensors/microcontroller device).

6.3. Scalability

Our current implementation is operational but is not scalable. Meanwhile, we may relatively easily reconfigure and deploy the architecture in a more distributed setting. For example, PlugAndPlayAgent could be evolved into a hierarchical architecture mapped for network subdomains.

In addition, QuantifiedFruitAgent could directly run on the microcontroller¹.

6.4. Collaboration

A future obvious direction is in making similar quantified things collaborative. In the case of bananas or fruits, various quantified things could exchange and share (and integrate) various experiences and data in order to extend and refine the analysis and predictive abilities. We can imagine scenarios for the fruit farms (plantation and conditioning), storage and delivery facilities.

Note that, in addition of offering predictions about fruit lifespan, this system could be adapted to provide other kinds of predictions, such as the percentage of fruit production that could be lost under specific transportation conditions.

6.5. From Prediction to Decision

Besides making predictions, this tool could also be extended to make suggestions and act on its own. For example, the device could make suggestions for temperature changes in real time. If we added a cooler device to current system, it could autonomously adjust temperature. Note that, as was explained in § 6.2, a preliminary version of our architecture has already also be used for control [16].

7. Conclusion

In this paper we have described a multi-agent architecture of quantified fruits for self-predicting maturation of fruits. It includes 6 types of agents, among them: a self-controller equipped with various sensors measuring storage conditions (light, temperature, humidity, etc.) and a self-prediction module based on a neural network. Our current implementation uses an Arduino microcontroller board with 5 sensors. We have implemented the architecture and have conducted various experiments with real settings and real data (storing bananas in diverse settings: room, refrigerator, in a box, with other fruits, etc.).

We believe these preliminary results to be promising. We think that they open the way for more experiments in testing the architecture in a more distributed and collaborative settings (various fruits and various stages of storing). We hope this tool may lead to improvements both in transportation methods by distributors, consumers and retailers, as well as their storage patterns and practices (refrigeration, packaging, etc.). Last, we believe this prototype architecture could be adjusted for other types of applications, such as collaborative monitoring of city air pollution (e.g., by citizens riding bicycles equipped with such architectures).

¹Nonetheless, JADE can only be used to implement agents to execute in Java-compatible systems. Thus, a JADE agent cannot (yet) be directly deployed on an Arduino board since it currently only provides support for development of C programs. Therefore, we have created a software layer to interface JADE with Arduino boards via sockets.

Acknowledgements to CAPES, CNPq and FAPERJ for their support through scholarships and fellowships.

References

- [1] M. Swan, "Sensor mania! The Internet of Things, wearable computing, objective metrics, and the Quantified Self 2.0," *Journal of Sensor and Actuator Networks*, vol. 1, no. 3, pp. 217–253, 2012.
- [2] N. M. Nascimento, C. J. Lucena, and H. Fuks, "Modeling quantified things using a multi-agent system," in *IEEE / WIC / ACM International Conference on Web Intelligence and Intelligent Agent Technology (WI-IAT)*. IEEE, 2015, pp. 26–32.
- [3] M. Swan, "Connected car: Quantified self becomes quantified car," *Journal of Sensor and Actuator Networks*, vol. 4, no. 1, pp. 2–29, 2015.
- [4] D. Johnson, N. Hipps, and S. Hails, "Helping consumers reduce fruit and vegetable waste: Final report," Waste and Resources Action Programme (WRAP), U.K., Tech. Rep., 2008.
- [5] A. Boe and D. Salunkhe, "Ripening tomatoes: Ethylene, oxygen, and light treatments," *Economic Botany*, vol. 21, no. 4, pp. 312–319, 1967.
- [6] R. Khan, S. U. Khan, R. Zaheer, and S. Khan, "Future internet: The Internet of Things architecture, possible applications and key challenges," in *Frontiers of Information Technology (FIT), 2012 10th International Conference on*. IEEE, 2012, pp. 257–260.
- [7] J. Burrell, T. Brooke, and R. Beckwith, "Vineyard computing: Sensor networks in agricultural production," *Pervasive Computing, IEEE*, vol. 3, no. 1, pp. 38–45, 2004.
- [8] W. Lang, R. Jedermann, D. Mrugala, A. Jabbari, B. Krieg-Bruuckner, and K. Schill, "The "Intelligent Container" - A cognitive sensor network for transport management," *Sensors Journal, IEEE*, vol. 11, no. 3, pp. 688–698, 2011.
- [9] F. Kawsar, T. Nakajima, J. H. Park, and S.-S. Yeo, "Design and implementation of a framework for building distributed smart object systems," *The Journal of Supercomputing*, vol. 54, no. 1, pp. 4–28, October 2010.
- [10] G. Fortino, A. Guerrieri, W. Russo, and C. Savaglio, "Integration of agent-based and Cloud Computing for the smart objects-oriented IoT," in *IEEE 18th International Conference on Computer Supported Cooperative Work in Design (CSCWD'2014)*, May 2014, pp. 493–498.
- [11] —, "Middlewares for smart objects and smart environments: Overview and comparison," in *Internet of Things Based on Smart Objects – Technology, Middleware and Applications*. Springer, 2014, pp. 1–27.
- [12] F. Bellifemine, F. Bergenti, G. Caire, and A. Poggi, *Jade – A Java Agent Development Framework*. Springer, 2005, pp. 125–147.
- [13] Arduino, "Arduino," <http://www.arduino.cc/>.
- [14] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Nature*, vol. 323, no. 6088, pp. 533–536, 1986.
- [15] J. Newmarch, *Foundations of Jini 2 Programming*. Apress, 2007.
- [16] N. M. Nascimento, "FIoT: An agent-based framework for self-adaptive and self-organizing internet of things applications," Master's thesis, PUC-Rio, Rio de Janeiro, Brazil, August 2015.

Quality Assurance for Big Data Application– Issues, Challenges, and Needs

Chuanqi Tao

Computer Science&Engineering Department
Nanjing University of Science and Technology
Nanjing, China
taochuanqi@njjust.edu.cn

Jerry Gao

Computer Engineering Department
San Jose State University, San Jose, USA
Taiyuan University of Technology, Taiyuan, China
Corresponding to: jerry.gao@sjsu.edu

Abstract—With the fast advance of big data technology and analytics solutions, building high-quality big data computing services in different application domains is becoming a very hot research and application topic among academic and industry communities, and government agencies. Therefore, big data based applications are widely-used currently, such as recommendation, predication, and decision system. Nevertheless, there are increasing quality problems resulting in erroneous testing costs in enterprises and businesses. Current research work seldom discusses how to effectively validate big data applications to assure system quality. This paper focuses on big data system validation and quality assurance, and includes informative discussions about essential quality parameters, primary focuses, and validation process. Moreover, the paper discusses potential testing methods for big data application systems. Furthermore, the primary issues, challenges, and needs in testing big data application are presented.

Keywords— *Quality assurance, big data application quality assurance, big data validation.*

I. INTRODUCTION

According to IDC [1], the Big Data technology market will "grow at a 27% compound annual growth rate (CAGR) to \$32.4 billion through 2017". Today, with the fast advance of big data science and analytics technologies, diverse data mining solutions, machine learning algorithms, open-source platforms & tools, and big data database technologies have been developed, and become available to be used for big data applications. This suggests that big data computing and application services bring large-scale business requirements and demands in people's daily life. Big data-based application system is widely-used nowadays, such as recommendation system, predictions, recognized patterns, statistical report applications, etc. Emergent big data computing and services can be used in many disciplines and diverse applications, including business management, library science, energy and environment, education, biomedical, healthcare and life science, social media and networking, smart city and travel, and transportation, etc.[2]. Nevertheless, due to the huge volume of generated data, the fast velocity of arriving data, and the large variety of heterogeneous data, the big data based applications brings new challenges and issues for QA engineers. For instance, it is a hard job to validate the correctness of a big data-based prediction system due to the large scale data size and the feature of timeliness. Therefore, Big data quality validation and big data-based application system quality assurance becomes a critical concern and research subject. Although there has been a numerous of published papers [2-6] addressing data quality and data quality assurance in the past, seldom researches focus on validation for big data application quality. There is an emergent need in research

work to quality study issues and quality assurance solutions for big data applications.

This paper is written to provide our perspective views on big data system validation for quality assurance. The paper is organized as follows. Section II discusses the typical types of big data systems and covers the essential quality parameters and their associated factors. Section III reviews and compares the existing testing methods for big data system validation. The major issues, challenges, and needs are presented in Section IV. Conclusions are in Section V.

II. UNDERSTANDING QUALITY ASSURANCE FOR BIG DATA APPLICATION SYSTEM

This section discusses the scope and process of quality assurance for big data application systems. Moreover, it covers the primary quality parameters with related factors.

Big data applications have the following unique features: (a) statistical computation based on multi-dimensional large-scale data sets, b) machine-learning and knowledge based system evolution, c) intelligent decision making with uncertainty, d) non-oracle functions, and e) complicated visualization. These unique features bring more interesting quality assurance and QoS requirements, challenges, and needs. Based on the recent feedbacks from engineers at Silicon Valley, how to assure the quality of big data-based application systems becomes a critical concern and research subject currently.



Figure 1 The Typical Types of Big Data Application Systems

A. Scope and Process of Big Data Application Quality Assurance

Big data applications provide services for prediction, recommendations, decisions support through large-scale data sets and complicated intelligent algorithms. Figure 1 describes the typical types of big data applications.

In general, *big data application quality assurance* refers to the study and application of various assurance processes, methods, standards, criteria, and systems to ensure the quality of big data system in terms of a set of quality parameters. Figure 2 shows a sample scope of validation for quality assurance of big data applications.



Figure 2 The Scope of Validation for Big Data Application System Quality

Compared to conventional software testing, a test process of big data based applications primarily focuses on their unique features, such as oracle problems, learning capability, and timeliness testing. Figure 3 shows a sample test process (function testing) for big data application system validation.

The testing process, shown in Figure 3, includes the following steps.

Step1 Big data system function testing, including rich oracles, intelligent algorithms, learning capability, as well as domain-specific functions;

Step 2 Big data system function testing, including system consistency, security, robustness, and QoS;

Step 3 Big data system feature testing, checks usability, system evolution, visualization, and so on;

Step 4 Big data system timeliness testing, targets time related feature testing, including continuous testing, real-time testing, life-time testing, and others.

B. Quality factors for big data application validation

Conventional system quality parameters such as performance, robustness, security, etc., can be applicable onto big data systems. They are listed below.

- **System Performance** –This parameter indicates the performance of the system, such as availability, response time, throughout, scalability, etc.
- **System Data Security** –This parameter could be used to evaluate the security of big data based system in

different perspectives. Using this parameter, data security could be evaluated in various perspectives at the different levels.

- **System Reliability** –This parameter is used to evaluate the durability of the system when performing a required function under stated conditions for a specified period of time.
- **System Robustness** - This parameter evaluates the ability of a system to resist change without adapting its initial stable configuration.

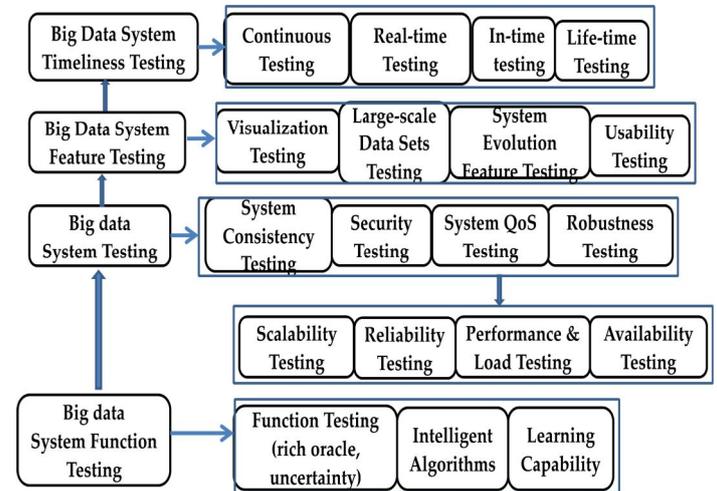


Figure 3 A Quality Test Process for Big Data Application System

In addition, due to the special characteristics, such as oracle problems, big data applications bring some impacted factors contributing to the challenges of system quality assurance. There are two typical big data applications: a) recommendation systems, and b) prediction systems. We have collected a number of common quality parameters from survey. Figure 4 summarizes the typical quality factors for prediction and recommendation systems in a fishbone graph respectively. Those factors are presented in taxonomy below.

Quality factors for prediction systems

- **System Correctness, which is a quality factor** used to evaluate the correctness of the big data applications. Unlike the conventional system, big data applications are hard to validate their correctness. For instance, prediction-related software is mainly developed to make predictions or better understand about real world activities. Hence, it is difficult to determine the correct output for those types of software. Correctness is related to the prediction pattern or model. For instance, some models are more likely used to predict point of inflexion values while some other models are doing well in predicting continuity. Thus, in order to verify the correctness of the system effectively, engineers need to evaluate the capability of prediction in the specified conditions and environments.
- **System Accuracy, which is used to evaluate** if the system yields true (no systematic errors), and consistent (no random errors) results. Some big data applications

are developed to find previously unknown answers, thereby only approximate solutions might be available. This can be called uncontrollable prediction. Some prediction is used to prevent something happening in

the future, and the prediction result will affect actions or behaviors. In turn, those actions can promote the prediction result.

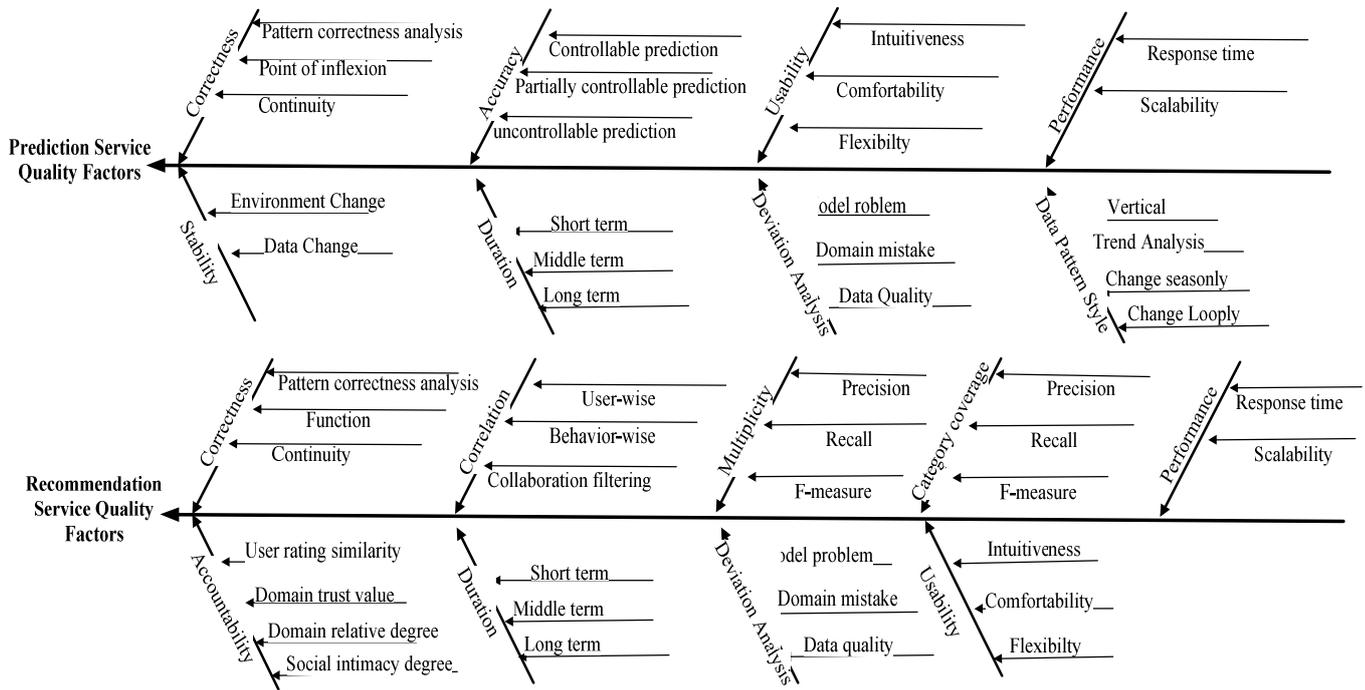


Figure 4 Big Data Application System Quality Factors

- **System Stability, which** reflects the stability of the system prediction while environment change or data changes. For example, if the prediction capability of a system is stable with little changes when statistical data are acquired from different timeframes.
- **System Consistency, which** is a quality indicator useful to evaluate the consistency of the targeted system in different perspectives. Due to the inherent uncertainties in system models, some applications do not produce single correct output for a given set of inputs. This leads to hardly determining the expected behaviors of the software. In such situation, domain-specific experts could provide opinions to support system consistency.
- **Duration, which** indicates the expected prediction period. It can measure how up-to-date data is, and whether it is correct despite the possibility of modifications or changes that impact time and date values [6]. For instance, commonly-used prediction duration in enterprise management can be divided into short term, middle term, and long term.
- **Deviation Analysis, which** is used to analyze the prediction deviation within an accepted range or confidence interval.
- **System usability, which** is a parameter that indicates how well the big data application service can be used. This can be very subjective due to different developers

and users have diverse user experiences. The typical usability factors include intuitiveness, comfortability, and flexibility.

- **System Performance, which** is a distinct quality factor for big data application service. It is useful to evaluate how well big data are structured, designed, collected, generated, stored, and managed to support large-scale prediction services.

Quality factors for recommendation systems

- **Correctness** – This quality factor reflects if the recommended service or commodity meets the demands of customers. Correctness could be subjective between different persons. Thus, how to measure correctness is still a challenge for quality assurance engineers.
- **Correlation** – This quality factor evaluates the degree of correlation of the recommended service. This involves various recommendation strategies, such as user content-based, behavior-based, and collaboration filtering-based.
- **Multiplicity** – This quality factor refers to the measurements for repeatability of recommended service. For instance, a poor quality system probably recommends too many repeated or similar commodities to users.

- **Category Coverage** – This indicator is useful to evaluate the coverage rate for diverse categories. This factor measures the completeness of recommendation within a selected domain.
- **Accountability** –This quality parameter is very important and mandatory for both big data service applications and users. This could be measured in a quantitative way, such as user rating similarity, domain trust value, domain related degree, and social intimacy degree.
- **Duration** –This factor indicates the expected recommendation period. For instance, commonly-used recommendation duration in enterprise management can be divided into short term, middle term, and long term.
- **Deviation Analysis** –This factor is used to analyze the recommendation deviation within accepted range or confidence interval.
- **System usability** –This parameter indicates how well big data application service can be used. This can be very subjective due to different developers and users have diverse user experiences.
- **System Performance**–This is a distinct quality factor for big data application service, and it is useful to evaluate how well big data are structured, designed, collected, generated, stored, and managed to support large-scale recommendation services.

In addition to the two typical applications discussed above, there are more big data related applications such as machine learning system, ranking system, and search system. Due to page limits, we do not list their quality factors here. A comparison of conventional testing and big data application testing in detail is presented in Table 1.

Table 1 A comparison of conventional testing and big data application testing

	Conventional Testing	Big Data Application Testing
Primary Objectives	- Validate the quality of software, including functions, programs, performance, etc.	- Provide on-demand testing services for big data application systems to support software validation and quality engineering process.
Testing Focuses	- Diverse software errors in its structures, functions, behaviors, user interfaces, and connections to the external systems. - System non-functional requirements such as performances, reliability, availability, vertical scalability, security, and etc.	- Non-oracle problem or rich oracle function problem. - Complicated algorithms. - Large-scale data input. - Complicated data models and integrations.
Test Input	- Limited scale - Specified data formats - Structured data	- Large-scale data volume with diverse formats and media - Structured and non-structured data - Timeliness
Testing Execution	- Offline testing in a test lab before product delivery. - Testing in a cloud-based test environment.	- On-demand test execution in a cloud-based virtual test environment. - Continuous testing for big data applications.
Test Coverage	- Function-based, data flow, structure-based, state diagram.	- To be developed; lack able currently.
Data Model	- Data partition, boundary analysis, etc.	- Training data; sampling data; classifier; image data classification; pseudo-oracles
Testing Environment	- A pre-configured test environment in a test lab with purchased hardware/software and tools.	- Provide testing environment references - Develop rapid reuse framework
Testing Process	- Enterprise-oriented test processes for each project.	- Crowd sourcing-based process - Learning-based testing - Classification based testing
Testing Techniques	- Apply selected well-known white-box and black-box testing techniques at the component level (or unit level) and the system level.	- Required innovative continuous, timeliness, and currency testing techniques. - New testing solutions to deal with multi-dimensional large-scale data sets, uncertainty data, learning-based system evolution, and complicated visualization.
Testing Tools	- Use limited testing solutions and tools with the purchased licenses. - Select and use diverse testing tools solutions which are pre-configured, installed, and deployed.	- Support development process. - Construct the whole process tool chains. - Data analysis tool. - Continuous evaluation including crowdsourcing and sampling.
Tool Connectivity and Platform	- Traditional test tool/solution integration and composition.	- Domain-specific application. - On-demand selective solutions which support users to integrate and composite test solutions and tools. - Incremental data sets.

III. VALIDATION METHODS FOR BIG DATA APPLICATION

This section discusses and reviews the existing research results in software testing methods which have been used to validate various types of big data applications including

intelligent systems, data mining programs, bioinformatics programs, and learning based applications.

Program-based software testing –Conventional program-based testing methods have been used in big data analytics applications. Csallner et al. presents a novel technique that

systematically searches for such bugs in MapReduce applications and generates corresponding test cases [18]. The technique works by encoding the high-level MapReduce correctness conditions as symbolic program constraints and checking them for the program under test. Shang et al. proposed an approach to uncover the different behaviors of the underlying platforms for BDA Apps using Hadoop between runs with small testing data and large real-life data in a cloud environment [19].

Classification-based testing—A classification approach to program testing usually involves two steps: a) training a classifier to distinguish failures from successful cases on a selected subset of results, and then b) applying the trained classifier to identify failures in the main set of results. A resembling reference model is usually used to train a classifier. More specifically, there are techniques for applying pattern classifications to alleviate the test oracle problems. Last et al. [9] and Vanmali et al. [11] apply a data mining approach to augment the incomplete specification of legacy systems. They train classifiers to learn the casual input-output relationships of a legacy system. Podgurski et al. classify failure cases into categories [10]. However, they do not study how to distinguish correct and failure behaviors of programs. Later, their research group further proposes classification tree approaches to refine the results obtained from classifiers [8]. Bowring et al. use a progressive machine learning approach to train a classifier on different software behaviors [7]. They apply their technique in the regression testing of a consecutive sequence of minor revisions of a program.

Metamorphic testing (MT) - This is a classic approach to testing programs that do not have oracles. Whenever a formal oracle is not available or costly to apply, we run into a *test oracle problem*. A test oracle is a mechanism against which testers can check the output of a program and decide whether it is correct. When an oracle is not available, other means of determining whether the test result is correct are known as *pseudo-oracles*. MT operates by checking whether a program under test behaves according to an expected set of properties known as metamorphic relations. A metamorphic relation specifies how a particular change to the input of the program should change the output. MT was used for testing scientific applications in different areas such as machine learning applications [12, 13], bioinformatics programs [14], programs solving partial differential equations [15] and image processing applications [16]. When testing programs solving partial differential equations, MT uncovered faults that cannot be uncovered by special value testing [15].

Learning-based testing – This involves how to adopt the various learning approaches and mechanisms to support testing for big data systems. Meinke et al. developed a technique for automatic test case generation for numerical software based on learning based testing (LBT) [17]. The authors first created a polynomial model as an abstraction of the program under test. Then the test cases are generated by applying a satisfiability algorithm to the learned model.

Crowd-sourced testing—This testing approach uses freelance testers and/or contracted engineers in a crowd

sourcing community. It is a cost-effective method to validate a machine-learning based application systems, such as a human face recognition system. Currently, crowd-sourced testing has been used in mobile app testing and mobile TaaS (Testing as a Service). One good example is uTest (<http://www.utest.com/company>).

Data model-based testing – Since big data are the input values for big data application systems, diverse data models can be used to assist test case generations. Vilkomir et al. presents a method to automatically generate test cases for a scientific program having many input parameters with dependencies [20]. They use a directed graph to model the input data space, including parameters and values as well as their dependencies. Valid test cases can be automatically generated based on the directed graph model. Since their model satisfies the probability law of Markov chains, it can be used to generate random and weighted test cases according to the likelihood of taking the parameter values.

Rule-based software testing – This approach could be used in testing rule-based or knowledge based systems. The basic idea is to design test cases based on the rules specified in an expert system. Deason et al. in [21] proposed a rule-based test data generation method for Ada programs. They demonstrated that rule-based test data generation is feasible. The paper shows a great promise in assisting test engineers in test generation. Andrews et al. presented a test pattern generation approach based on VHDL specific heuristic rules [22]. Their results indicated the rule-based approach leading to a better test coverage.

Conventional black-box software testing – To assure the system performance and other related QoS parameters of big data applications, engineers could use convention black-box approaches to controlling their quality. Typical examples include decision table testing, equivalence partitioning, boundary value analysis, cause-effect graph, and use case testing, and so on.

IV. ISSUES, CHALLENGES, AND NEEDS

There are a number of major issues and challenges in big data quality validation and assurance. Here are typical ones.

Issue #1—*What are the adequate test models and test coverage criteria for big data based service applications?*

With the fast advance of big data technologies and analytics methods, more and more big data based applications and service systems are developed to be used in many areas of our daily life, including smart cars, smart city, business intelligence, environmental control, and so on. The increasing deployment of big data applications and services raises quality assurance concerns. In the past, many existing white-box and black-box software test models and adequate validation criteria are developed to address validation needs of software applications in functions, behaviors, and structures. However, these existing adequate test models only focus on program functions, state-based behaviors, and program structures. In the software testing and quality assurance community, there is a lack of research work on adequate test modeling and

coverage analysis for big data application systems by considering their special features and needs in rich oracle functions, machine-learning based system evolutions, knowledge based system intelligence, and multi-dimensional large-scale data sets.

Hence, according to real world practitioners, there is a clear demand on establishing well-defined test coverage criteria for big data application systems. Otherwise, engineers and big data analysts will have difficult time to figure out when they should stop quality testing for big data applications. This leads to the first demand described below.

Need #1–Developing well-defined adequate validation models and criteria to address the special features and needs of big data applications and services.

Issue #2 –Where are the well-defined big data system quality assurance programs and standards, including processes, assessment metrics, regulations, and policies?

As we discussed in [25], ISO is working on updating of existing data quality assurance standards and programs for big data quality assurance. Considering the popularity of big data applications and services, we must address the quality control and assurance of big data based applications. Here, we point out the second emergent need below.

Need #2 – Establishing quality assurance programs and standards to consider the special QoS parameters and factors of big data applications and services to ensure system quality.

Issue #3– What are test automation solutions and tools supporting efficient and large-scale testing operations for big data applications and services?

In the past three decades, many test automation tools and solutions have been developed for engineers in assisting test automation activities and operations. Unfortunately, most of these tools are only useful to validate software system functions, behaviors, program structures, and system performance and other QoS parameters. As discussed in section III, there are a few published papers addressing validation methods.

However, these validation methods are not designed and developed to address the special features and needs of big data application systems. As discussed in Section III, there has been a few of published research work addressing special validation needs in big data based applications. However, there is a clear lack of research work on automatic validation methods and solutions for big data application services. Therefore, the third emergent need for big data applications is listed below.

Need #3 –More innovative adequate testing methods and test automation tools to address the special needs and features of big data application systems and services.

Unlike conventional software test automation tools, these expected test automation solutions must consider big data applications' special features listed below:

- Large-scale big data inputs with diverse formats, and structured and non-structured data;
- Learning and knowledge based system evolutions;
- Non-oracles problems and rich oracle functions with uncertainty;
- New QoS parameters, such as accuracy, accountability, usability, and
- Data modeling

V. CONCLUSIONS

With the fast advance of big data management technologies and analytics solutions, how to build high-quality big data application services becomes a very hot subject. Nevertheless, there are increasing quality problems resulting in erroneous data costs in enterprises and businesses [25]. Current research work seldom discusses how to effectively validate big data applications to ensure system quality. This paper provides informative discussions on big data system validation and quality assurance, including the essential concepts, focuses, and validation process. Moreover, the paper identifies and discusses some primary quality factors. In addition, it presents a comparison between conventional testing and big data application testing. Furthermore, the primary issues, challenges, and needs are presented.

REFERENCES

- [1] Editor of Hosting Journalist, IDC forecast: big Data technology and services to hit \$32.4 billion in 2017, December 18, 2013.
- [2] A.O. Mohammed, S.A. Talab. Enhanced extraction clinical data technique to improve data quality in clinical data warehouse. International Journal of Database Theory and Application, 8(3): 333-342, 2015.
- [3] M. R. Wigan, R. Clake. Big data's big unintended consequences. IEEE Computer, 46 (6):46-53, 2013.
- [4] J. Alferes, P.Poirier, C. Lamaire-Chad, et al. Data quality assurance in monitoring of wastewater quality: Univariate on-line and off-line methods. In Proc. of the 11th IWA conference on instrumentation control and automation, pp. 18-20, September, 2013.
- [5] R. Clarke. Quality factors in big data and big data analytics. Xamax Consultancy Pty Ltd. 2014.
- [6] A. Immonen, P. Paakkonen, and E. Ovaska. Evaluating the quality of social media data in big data architecture. IEEE Access, 3: 2028 - 2043 October 16, 2015.
- [7] J.F. Bowring, J.M. Rehg, and M.J. Harrold. Active learning for automatic classification of software behavior. In Proc. of the 2004 ACM SIGSOFT International Symposium on Software Testing and Analysis (ISSTA). ACM, New York, NY, pp.195–205, 2004.
- [8] P. Francis, D. Leon, M. Minch, and A. Podgurski. Tree-based methods for classifying software failures. In Proc. of the 15th International Symposium on Software Reliability Engineering (ISSRE). Los Alamitos, CA, pp. 451–462. 2004.
- [9] M. Last, M. Friedman, and A. Kandel. The data mining approach to automated software testing. In Proc. of the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD). ACM, New York, NY, pp. 388–396, 2003.

- [10] A. Podgurski, D. Leon, P. Francis, W. Masri, M. Minch, J. Sun, and B. Wang. Automated support for classifying software failure reports. In Proc. of the 25th International Conference on Software Engineering (ICSE), LosAlamitos, CA, pp. 465–475, 2003.
- [11] M. Vanmali, M. Last, and A. Kandel. Using a neural network in the software testing process. *International Journal of Intelligent Systems*, 17 (1): 45–62, 2002.
- [12] X. Xie, J.W. Ho, C. Murphy, G. Kaiser, B. Xu, and T.Y. Chen. Testing and validating machine learning classifiers by metamorphic testing. *Journal of System and Software*, 84 (4):544–558, 2011.
- [13] C. Murphy, G. Kaiser, L. Hu, and L. Wu. Properties of machine learning applications for use in metamorphic testing. In Proc. of the 20th International Conference on Software Engineering and Knowledge Engineering (SEKE), pp.867–872, 2008.
- [14] T.Y. Chen, J.W.K. Ho, H. Liu, and X. Xie. An innovative approach for testing bioinformatics programs using metamorphic testing, *BMC Bioinform.* 10(2009).
- [15] T. Chen, J. Feng, and T.H. Tse. Metamorphic testing of programs on partial differential equations: a case study. In Proc. of the 26th Annual International Computer Software and Applications Conference, (COMPSAC), pp. 327–333, 2002.
- [16] J. Mayer, R. Guderlei. On random testing of image processing applications, In Proc. of the 6th International Conference on Quality Software (QSIC), pp. 85–92, 2006.
- [17] K. Meinke, F. Niu. A learning-based approach to unit testing of numerical software, in: A. Petrenko, A. Simo, J. Maldonado (Eds.), *Testing Software and Systems*, Lecture Notes in Computer Science, vol. 6435, Springer, Berlin, Heidelberg, 2010, pp. 221–235.
- [18] C. Csallner, L. Fegaras, C. Li. New Ideas Track: Testing MapReduce-style programs. In Proc. of 9th ACM SIGSOFT Symposium on the Foundations of Software Engineering (ESEC/FSE’11), pp 1-4, 2011.
- [19] W. Shang, Z.M. Jiang, H. Hemmati, B. Adams, and A.E. Hassan. Assisting developers of big data analytics applications when deploying on Hadoop clouds. In Proc. of 35th International Conference on Software Engineering (ICSE), pp 402-411, 2013.
- [20] S.A. Vilkomir, W.T. Swain, J.H. Poore, and K.T. Clarno. Modeling input space for testing scientific computational software: a case study. In Proc. of the 8th International Conference on Computational Science, Part III (ICCS), pp. 291–300, 2008.
- [21] W.H. Deason, D.B. Brown, and K.H. Chang. A rule-based software test data generator. *IEEE Transactions on Knowledge and Data Engineering*, 3(1): 108-117, 1991.
- [22] A. Andrews, A.O. Fallon, and T. Chen. A Rule-Based Software Testing Method for VHDL Models. *VLSI-SOC 2003*: 92.
- [23] W. Afzal, R. Torkar, and R. Feldt. A systematic review of search-based testing for non-functional system properties. *Information and Software Technology*, 51 (6):957–976, 2009.
- [24] T. Clune, R. Rood, Software testing and verification in climate model development, *IEEE Software*, 28 (6):49–55, 2011.
- [25] J. Gao, C.L. Xie, and C.Q. Tao. Quality assurance for big data—issues, challenges, and needs. In Proc. of IEEE 9th International Symposium on Service oriented System Engineering, OZFORD, UK, 2016.

ACKNOWLEDGEMENT

This paper is supported by the National Natural Science Foundation of China under Grant No.61402229 and No.61502233; the Open Fund of the State Key Laboratory for Novel Software Technology (KFKT2015B10), and the Postdoctoral Fund of Jiangsu Province under Grant No.1401043B.

Interactive Tool for Iterative Test Suite Construction

Matthew Patrick
Department of Plant Sciences
University of Cambridge
United Kingdom
Email: mtp33@cam.ac.uk

Abstract—We can only test software effectively if we understand how it is intended to behave. For some categories of programs, such as scientific models, it is not obvious what the output of the software should be. New techniques are needed to help domain experts, such as scientists, gather the knowledge they need to construct suitable tests and oracles. This paper introduces a new interactive tool for iterative test suite construction that is based upon the scientific method paradigm that scientists are familiar with. We apply our technique to a deterministic mathematical model, used to predict the spread of disease, and show how it helps scientists uncover situations they had not yet considered. Of the 15 hypotheses originally created by modellers, our technique found discrepancies in all but one, allowing us to refine them into a more rigorous test suite.

I. INTRODUCTION

The Human Oracle Problem [1] can impose a significant obstruction to the quality of software testing. It is relatively easy to produce a large number of test cases using automated techniques, but the effort required to determine whether each of their outputs is correct can be prohibitively expensive. An oracle is a mechanism by which it is determined for each input what the output should be. Testing techniques typically assume the availability of an automated oracle [1]. Yet, many programs belong to a category of software which are said to be ‘non-testable’ [2]. This means it is just as expensive to develop an automated oracle (and just as hard to make sure it is correct), as it is the software we are testing. The end result is that human experts have to act as the oracle instead, evaluating each output one at a time, to assess if it is correct.

In addition to problems with the throughput of test output evaluations, there are also issues related to accuracy. One research group had to retract five papers from top level journals, such as Science, because its software contained a fault that remained unnoticed [3]. It was later learnt the protein crystal structures they were investigating were being inverted. Similarly, nine packages for seismic data processing were found to produce significantly different results due to problems such as off-by-one errors [4]. The predictions made from the packages would have led people using them to come to different conclusions, potentially leading to \$20 million oil wells being drilled in the wrong place. We cannot just rely on experts to look at each output and check it matches their expectation. This approach is likely to miss important errors, since the output may appear reasonable and still be incorrect.

Scientific software is particularly difficult to test, because its correct behaviour is not normally known in advance. The software is developed to investigate new research questions and the perceptions of researchers change as their hypotheses are explored [5]. If we already knew the answers to these questions, there would be no need to create the software in the first place. Intrinsic difficulties occur in evaluating the output of scientific software, due to the stochasticity and uncertainties in the underlying model. Scientific software frequently incorporates complex models with nonlinear dynamics that are difficult to test. Each numerical approximation has the potential to introduce new errors [5], inaccuracies may arise due to the way in which experimental data are collected, and then when the model is implemented on a finite precision computer, some further accuracy is inevitably lost.

In this paper, we address the Human Oracle Problem by adapting the well-known strategy of Iterative Hypothesis Testing to allow domain experts to produce stronger and more reliable oracles for test suites. Iterative Hypothesis Testing is a core component of the scientific research method. Scientists start with initial hypotheses about how the system they are studying is expected to behave. Then, as these hypotheses are tested and explored, new information is gathered that can be used to add to and refine them [6]. Similarly, scientists might have some initial hypotheses as to what the outputs of the software they are developing should be, but they do not at first have sufficient information to create a rigorous set of test cases. Instead, Iterative Hypothesis Testing can be used to find cases in which these hypotheses do not hold, and identify ways in which they can be improved through repeated refinements.

Since our technique operates in a form scientists are already familiar with, it is highly suitable for testing scientific software (it can also be used on a wide range of other software). We apply our tool to an implementation of an epidemiological model. Hypotheses are generated in consultation with epidemiological modellers and then refined using our tool. The rest of this paper is arranged as follows: Section II gives more details about our testing technique and Section III describes the model we are testing; Section IV presents our initial hypotheses and Section V explores the results of applying Iterative Hypothesis Testing to them; Section VI presents some related work; Section VII provides our conclusions and Section VIII describes some opportunities for further research.

II. OUR INTERACTIVE TEST SUITE CONSTRUCTION TOOL

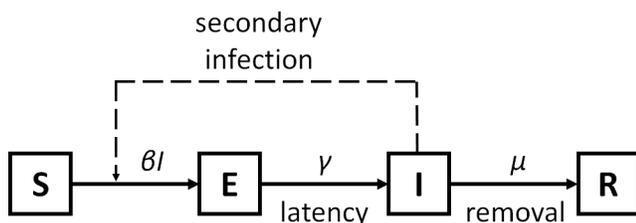
We introduce a new tool that helps domain experts iteratively refine their test suites. An interface is provided in which hypotheses can be entered as predicate relations about the output values produced for particular sets of inputs. The interface can also be used to define relations which describe how the output should change if the input is modified. These ‘metamorphic relations’ are an established way to test software for which an oracle is not available [7]. However, our tool differs from previous research by enabling the iterative refinement of these relations through a search-based approach.

Our tool uses random testing to identify discrepancies in the output between the values expected by the hypotheses and the values actually produced. It then performs a directed search to identify conditions under which the discrepancies occur. Our tool then proposes a slightly modified form of the hypotheses to address these discrepancies. For example, suppose we incorrectly assumed that one of the outputs of our software was always less than 10. Our technique uses random testing to identify cases in which this does not hold, then explores the search space to conclude it is never greater than 11. Of course, there is no guarantee the new hypotheses will not fail, and they may need to be refined further.

Rather than simply trusting the new hypotheses proposed by our tool, it is often more effective to use it interactively. If our tool cannot find any discrepancies at first, we can increase the number of random test cases it generates and instruct the tool to consider boundary conditions (such as setting values to 0). When the tool indicates particular hypotheses and input/output values that fail, we can look back at the software and consider why this is happening. The information provided by the tool allows a domain expert to determine whether the discrepancy indicates a fault in the software, or if it is caused by a misunderstanding of how it should behave. This helps the expert decide if the software or hypotheses should be changed.

Our tool is built on top of QuickCheck for R [8]. We provide a wrapper interface that transforms hypotheses into parametrised unit tests. These are then tested using automatically generated random test cases. Random testing is a straightforward and inexpensive software testing technique [9]. It can generate a large number of input values in a short amount of time, then verify the results using automatic tests of our hypotheses. Despite its simplicity, random testing is often more effective than advanced testing techniques [10].

Fig. 1. SEIR Model schematic



III. CASE STUDY: THE SEIR MODEL

We evaluate our technique by applying it to an epidemiological SEIR model [11]. The SEIR model (see Figure 1) tracks the hosts of a pathogen through the following compartments: *Susceptible* (not infected), *Exposed* (infected but neither infectious nor showing symptoms), *Infectious* (infectious and showing symptoms) and *Removed* (no longer infectious, because they are dead or recovered). Hosts may start off susceptible (S), but when they are exposed (E) to infectious hosts, they enter into a latent period before being infectious (I); later the infectious hosts are removed (R) from the population.

We run tests with a fixed total amount of host ($N = 2000$); the amount of infectious host (I) is selected uniformly at random between 0 and $N/2$, and the remaining host is placed into the susceptible (S) compartment. These settings were designed to reflect the conditions under which new infectious material enters the population from outside the model. γ and μ are selected uniformly at random between 0 and 1, β between 0 and $1/N$. Each simulation is run using 100 test cases over 50 time units, with a discrete time step of 0.01 units.

IV. INITIAL HYPOTHESES

1) Sanity Checks:

- H1:** None of the compartments should ever contain a negative amount of host (it is biologically impossible)
- H2:** The total amount of host should not differ at each time step (our model assumes a closed population)
- H3:** The amount of susceptible host (S) should never increase (infected hosts do not become susceptible again)
- H4:** The amount of removed host (R) should never decrease (The R curve should be monotonically increasing)
- H5:** The exposed host (E) peak should not occur after the infectious host (I) peak (E acts as a buffer for S to I)

2) Metamorphic Relations:

- H6:** Increasing the infection rate (β) should increase the peak of E (by creating a build-up of host)
- H7:** Increasing the latent rate (γ) should reduce the time until the peak of I (by allowing host to move faster from E)
- H8:** Increasing μ should increase the final amount of host in S (by allowing host to move faster from I)
- H9:** Increasing β should decrease the final amount of host in S (by allowing host to move faster from S)
- H10:** Increasing the number of susceptible host (S_0) should increase the peak of I (more host to be infected)

3) Mathematical Derivations:

- H11:** I should be increasing when $\gamma E > \mu I$, otherwise it should be decreasing (from rate equation for I)
- H12:** If $I = E = 0$, the state of the model should not change (every term in the equations contains the value I or E)
- H13:** Exact analytical solutions are available when $\gamma = 0$ ($I_t = I_0 e^{-\mu t}$ and $S_t = S_0 e^{\frac{\beta I_0 (e^{-\mu t} - 1)}{\mu}}$)
- H14:** Exact analytical solutions are available when $\beta = 0$ ($I_t = \frac{e^{-\gamma t} (E_0 \gamma (e^{t(\gamma - \mu)} - 1) + I_0 (\gamma - \mu) e^{t(\gamma - \mu)})}{\gamma - \mu}$)
- H15:** A final size equation can determine the value of S when $t = \infty$ ($\ln(\frac{S_0}{S_\infty}) = \frac{\beta}{\mu} (N - S_\infty)$)

V. EXPLORING AND REFINING THE HYPOTHESES

We applied our interactive tool to explore and refine the initial hypotheses presented in Section IV. Our tool found discrepancies in all but one of the hypotheses (**H4**). In many cases, the hypotheses tests can be refined automatically, through simple changes (e.g. the introduction of an edge case or the selection of suitable tolerance thresholds). However, it is sometimes necessary for a domain expert to pick the most appropriate change. In this section, we consider some examples of how our tool was able to help refine the hypotheses.

A. Complexities of the Model

Some discrepancies occurred due to edge cases in the model behaviour, not taken into account by the initial hypotheses. It is likely these edge cases would be identified by domain experts, if allowed a sufficient amount of time. However, it is much easier to find these cases automatically using our tool. For example, in **H5** we reasoned that since host pass through the E compartment on their way to the I compartment, the peak of E would be before that of I in time. Our tool found a case in which this did not hold, using the following input parameters: $I_0 = 477$, $\beta = 4.95 \times 10^{-4}$, $\gamma = 0.285$, $\mu = 0.986$. The discrepancy can be seen in Figure 2: the peak of E does not come before the peak of I because I is monotonically decreasing (i.e. it has no peak). A similar problem occurs with **H7**: increasing γ cannot make the peak of I come earlier if it is already monotonically decreasing. These hypotheses tests were refined by adding exceptions for the edge cases.

In some cases, it is only possible to identify discrepancies in the hypotheses by using specific boundary values. Random testing is inefficient at finding these cases because it explores the input domain evenly, so our tool also selects values from the edge of the input domain. We found boundary tests to be particularly useful for the situations in which parameters or compartment values are zero. For example, when $S = 0$,

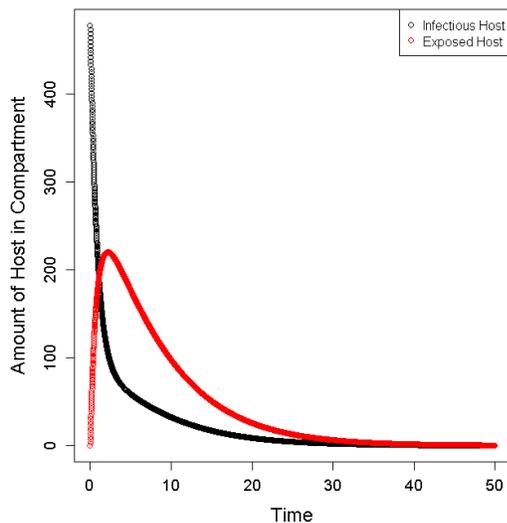


Fig. 2. Peak in E but not in I (**H5**)

changing the value of β has no effect on the peak of E , because there is no host to move into E from S . We therefore added an exception for this situation in **H6**. Similarly in **H8**, increasing μ has no effect on the final amount of host in S when $\beta = 0$ because host can never leave the S compartment. By finding and addressing these edge cases, our tool helps to refine the hypotheses tests and make them usable in practice.

B. Complexities of the Implementation

In addition to edge cases in the model, our tool can also be used to explore the differences between mathematical behaviour and the behaviour of the model when implemented on a finite precision computer. Disparities are addressed by introducing a tolerance threshold into each hypothesis test. The optimum value of the threshold depends on the hypothesis being tested. For example we found the optimum threshold for **H1** to be around 1×10^{-6} , whereas for **H2** it was around 2×10^{-8} . Our tool identifies the optimum threshold by incrementally increasing its value and counting the number of times the hypotheses fail. Hypotheses tests fail on every run if the threshold is set too low, but if it is set too high, the hypotheses will never fail (even if there is a fault in the software). Our tool therefore sets the threshold to the smallest value at which there are no failures during 100 runs.

Finite precision computation also has an effect due to the discrete time steps that are used. For example, our tool found that **H11** does not hold because it incorrectly identifies the points at which I transitions from increasing to decreasing (see Figure 3). Just before the transition point, $\gamma E < \mu I$, so I should decrease. However, our tool found the next value can be higher, because the mathematical minimum occurs between the time steps and then the value of I increases. The same problem occurs when calculating the transition point at the maximum of I . Using our tool, we were able to address this by only checking up until the point before the transition.

VI. RELATED WORK

There have been other attempts to help make the construction of test oracles easier. For example, Staats et al. [12] use mutation testing and Loyola et al. [13] use network centrality analysis as a predictive measure to rank variables for their ability to detect faults. These approaches are useful for guiding testers towards parts of the code that deserve greater consideration, but they are not designed to help scientists in the iterative process of refining their test suite.

Salari and Knupp [14] suggest a number of tests that are useful for scientific software. For example, the Method of Manufactured Solutions creates tests by solving a partial differential equation backwards, i.e. inverting the equation to determine the inputs needed to create the correct solution. They also proposed trend tests (varying the input parameters and checking the overall pattern), symmetry tests (e.g. changing the order of inputs and checking that the results are the same) and comparison tests (using pseudo-oracles) [14]. Our work differs from this in that in addition to proposing a set of tests, we provide a tool for automatically refining them.

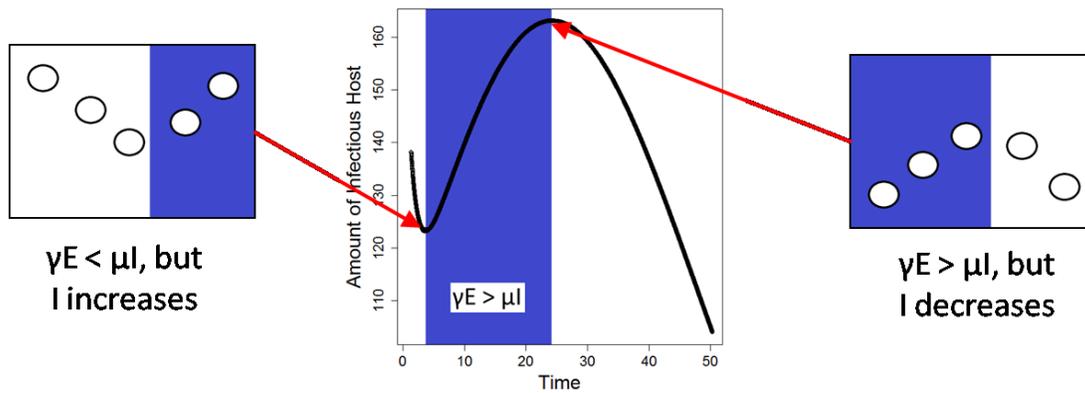


Fig. 3. Incorrectly Identified Increases and Decreases in I (H11)

VII. CONCLUSIONS

We presented a new interactive tool for iterative test suite construction that is easy for scientists to use because it is based on a methodology they are already familiar with (Iterative Hypothesis Testing). We have shown our tool can find discrepancies resulting from unforeseen complexities of the model and implementation. Although these issues might have been identified manually, if given enough time, it is faster and easier for scientists to create some initial hypotheses and then use our tool to identify the unexpected cases for which they need refining. The information gathered by our tool can be used to improve the tests through a mixture of automatic and interactive refinement. As a result, scientists can easily use our tool to create rigorous test suites for their software.

VIII. FURTHER WORK

We applied our tool to a simple model, so we could explain it more clearly. When discrepancies were found, it was relatively straightforward to identify the source of the error and update our hypotheses appropriately. However, this might not be so easy with complex models or other forms of scientific software. We need to make sure our tool can be used to identify the cause of each discrepancy, so we can respond efficiently. It would therefore be useful to evaluate our tool on other, more complex, software. We might also like to consider alternative strategies for exploring hypotheses, such as those based on advanced search-based software testing techniques or symbolic analysis of the differential equations.

Another issue arises over how we can know whether our approach is sufficiently rigorous. We will evaluate the effectiveness of metrics for coverage assessment, such as those based on control and data-flow criteria. It might be also be worthwhile to create new metrics specifically designed for scientific software, measuring coverage at the level of the experiments and hypotheses. Since scientific software recreates events that happen in the real world, we can also use techniques such as lab and field experiments to verify our results. We will investigate how best to use these in parallel with software development, to iteratively improve the tests.

IX. ACKNOWLEDGMENTS

This work was supported by the University of Cambridge/ Wellcome Trust Junior Interdisciplinary Fellowship “Making scientific software easier to understand, test and communicate through modern advances in software engineering”. We thank Richard Stutt and James Elderfield for useful discussions.

REFERENCES

- [1] E. T. Barr, M. Harman, P. McMinn, M. Shahbaz, and S. Yoo, “The Oracle Problem in Software Testing: A Survey,” *IEEE Trans. Software Engineering*, vol. 41, no. 5, pp. 507–525, 2015.
- [2] E. J. Weyuker, “On Testing Non-testable Programs,” *The Computer Journal*, vol. 25, no. 4, pp. 465–470, 1982.
- [3] Z. Merali, “Computational science: Error, why scientific programming does not compute,” *Nature*, vol. 467, no. 7317, 2010.
- [4] L. Hatton and A. Roberts, “How accurate is scientific software?” *Software Engineering, IEEE Transactions on*, vol. 20, no. 10, pp. 785–797, 1994.
- [5] J. C. Carver, M. S. Starkville, R. P. Kendall, S. E. Squires, and D. E. Post, “Software Development Environments for Scientific and Engineering Software: A Series of Case Studies,” in *Proc. 29th Int. Conf. Software Engineering*, 2007, pp. 550–559.
- [6] J. Hannay, C. MacLeod, J. Singer, H. Langtangen, D. Pfahl, and G. Wilson, “How Do Scientists Develop and Use Scientific Software?” in *Soft. Eng. for Computational Science and Eng., ICSE*, 2009. [Online]. Available: <http://dx.doi.org/10.1109/SECSE.2009.5069155>
- [7] Z. Q. Zhou, D. H. Huang, T. H. Tse, Z. Yang, H. Huang, and T. Y. Chen, “Metamorphic Testing and its Applications,” in *Proc. 8th Int. Symp. Future Software Technology*, 2004.
- [8] A. Piccolboni, “quickcheck,” 2015. [Online]. Available: <https://github.com/RevolutionAnalytics/quickcheck>
- [9] J. W. Duran, “An Evaluation of Random Testing,” *IEEE Trans. Software Engineering*, vol. 10, no. 4, pp. 438–444, 1984.
- [10] S. Shamsiri, J. M. Rojas, G. Fraser, and P. McMinn, “Random or Genetic Algorithm Search for Object-Oriented Test Suite Generation?” in *Proc. GECCO*, 2015, pp. 1367–1374.
- [11] M. Y. Li, J. R. Graef, L. Wang, and J. Karsai, “Global dynamics of a SEIR model with varying total population size,” *Mathematical Biosciences*, vol. 160, no. 2, pp. 191–213, 1999.
- [12] M. Staats, G. Gay, and M. P. E. Heimdahl, “Automated Oracle Creation Support, or: How I Learned to Stop Worrying about Fault Propagation and Love Mutation Testing,” in *Proc. 34th Int. Conf. Softw. Eng.*, 2012, pp. 870–880.
- [13] P. Loyola, M. Staats, I.-Y. Ko, and G. Rothermel, “Dodona: Automated Oracle Data Set Selection,” in *Proc. Int. Symp. Softw. Testing Anal.*, 2014, pp. 193–203.
- [14] K. Salari and P. Knupp, “Code Verification by the Method of Manufactured Solutions,” Sandia National Laboratories, Tech. Rep. SAND2000-1444, June 2000.

PRO-Fit: A personalized fitness assistant framework

Saumil Dharia, Vijesh Jain, Jvalant Patel, Jainikkumar Vora, Shaurya Chawla, Magdalini Eirinaki
Computer Engineering Department
San Jose State University
San Jose, CA, USA

ABSTRACT—*The advancements in wearable technology, where embedded accelerometers, gyroscopes and other sensors enable the users to actively monitor their activity have made it easier for individuals to pursue a healthy lifestyle. However, most of the existing applications expect continuous feedback from the end users and fail to engage those who have busy schedules, or are not as committed and self-motivated. In this work, we propose a framework that employs machine learning and recommendation algorithms in order to smartly track and identify user's activity by collecting accelerometer data, synchronizes with the user's calendar, and recommends personalized workout sessions based on the user's and similar users' past activities, their preferences, as well as their physical state and availability.*

KEYWORDS—*wearable technology, activity tracking, classification, recommendation, personalized assistant*

I. INTRODUCTION

The advancements in wearable technology, where embedded accelerometers, gyroscopes, GPS tracking and other sensors enable the users to actively monitor their activity have revolutionized the field, by allowing users to engage and track fitness activities. According to a study screening 200 existing health and fitness apps, the main priority for users is to have an application that makes any physical activity entertaining and rewarding, and motivates them to continue striving for achieving targets they set [1]. In this paper, we present PRO-Fit, a personalized fitness assistant framework. We are motivated by the fact that for busy individuals, the existing interactive models might not be enough to keep them motivated to engage in fitness-related activities. Apart from being a simple activity tracker, PRO-Fit automatically identifies when an individual can exercise, and proactively notifies the user. Instead of simple reminders, the framework automatically generates personalized fitness schedules, depending on the user's location, availability, and preferences.

The proposed framework incorporates two critical modules: an activity classifier, and a ranking and recommendation engine. We use machine learning algorithms on activity data to build predictive models that classify the user's activity into specific types. We build user profiles reflecting their current lifestyle (e.g. sedentary vs. active), age, weight, goals (e.g. time spent exercising each week), and preferences (e.g. favorite fitness activities, level of intensity etc.). This user profile is fed to a hybrid recommendation system that matches the user's profile to available activities, ranked in terms of similarity, but also taking into consideration the geo-location and time availability. For instance, PRO-Fit might recommend a 1-hour yoga class at the University fitness center during lunch time for

user A, who's employee at the University, and 20-min jogging at the nearest park for user B, who is a student and has 30 minutes between classes. The rest of the paper is organized as follows: In Section II we review the related work. In Section III we present the results of a user survey that helped us define the requirements for our framework. The architecture and implementation details are discussed in Section IV. In Section V we discuss some preliminary experimental results, and we conclude with our plans for future work in Section VI.

II. RELATED WORK

The existing activity trackers have inbuilt sensors that are able to recognize activity. A lot of research work has been done in this area, by employing machine learning algorithms on past user activity data [2], heart rate data [3], and accelerometer data [4,5, 6] to identify the type of activity, or estimate caloric consumption. In [7], the authors use an on-body chest sensor in coordination with a smartphone to collect the data for the activities performed by the individual, whether static or dynamic. In [8] the authors propose the Digital Fitness Connector (DFC) architecture, which allows the user to monitor physical activity in real-time as well as post-workout. The majority of the proposed approaches employ decision trees and their scalable variations (such as random forests) to perform the activity classification [9], however clustering approaches have also been used to split activity data into categories [10]. While most of existing works focus on improving the activity prediction process, most require extensive user profiling and interaction throughout the day. In our work, we propose a framework that minimizes the need for user input by *proactively* reminding him/her about their goals and generating personalized fitness recommendations based on their daily schedule and activities.

III. REQUIREMENTS ANALYSIS

In order to collect information about the users' preferences, devices/apps used, activities performed, feature they like the most, features to modify/add to the existing apps, activity schedule, we conducted a survey among more than 200 people¹. Based on the survey information we identified, among others, the following requirements: a) the motivation for most of the users is goal tracking, b) the favorite feature is to calculate the calories burnt and present the information on a daily basis, c) people try to schedule their fitness activities at least 2-4 times a week, for 1-2 hours each day, d) most people preferred working out alone, e) the activities that were mostly tracked were walking, jogging and cycling, f) people are

¹ The survey was conducted online, through different social networks. We received around 200 responses in 3 days. We kept the survey anonymous, except for the age (range between 18 and 45 years). The survey questions can be found at: <https://docs.google.com/forms/d/1MDpDs2yiSL8gYqddpAJX0jp4iFPg3tKe9bx3eZlgoj/viewform>

willing to plan their fitness schedule during work hours, g) a desirable feature would be a recommendation system for fitness activities, h) a desirable feature would be reminders and fitness session organizer. All the applications mentioned in Section II.B are more focused to track the user’s activities, but none of them provides a recommendation system that would help users to choose from activities based on their interest and accomplishment of goals. Based on these observations, we were motivated to design the PRO-Fit personalized fitness assistant framework that acts as a motivator and organizer for fitness activities making it easy for users to create and follow their workout plan and schedule the sessions according to their availability and preference.

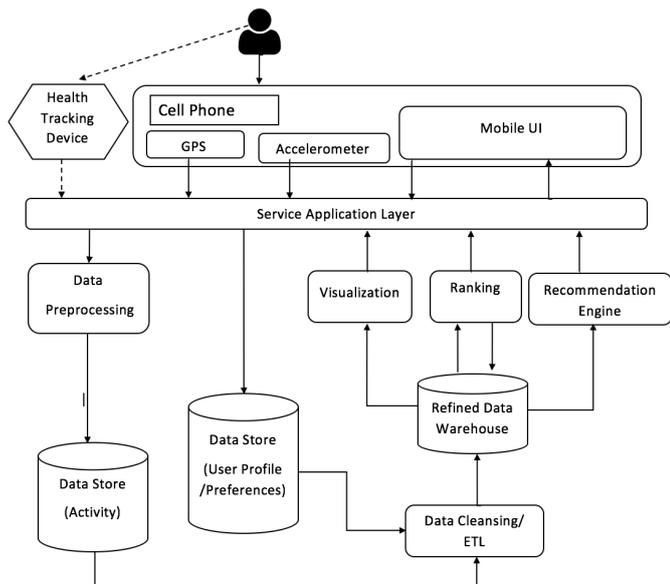


Figure 1. System Architecture Diagram

IV. ARCHITECTURE

The proposed framework architecture is shown in Figure 1. The framework is designed such that it scales for multiple clients. In what follows, we discuss the functionality and the most important design decisions for each module.

A. Health Tracking Device Client

The framework’s client is an Android/iOS mobile application, which contains UI screens and interacts with Mobile GPS and accelerometer to get user’s activity and location data. The end user interacts with the mobile app to sign up, occasionally give inputs and receive recommendations on fitness sessions. Third party wearable devices expose developer APIs to pull data for the analysis. Such devices can sync with the Application and service layer can pull data from respected data stores of wearable service providers.

B. Application Service Layer

This is the interface for all interactions of the mobile application to the outer world of the system. This layer can be divided into two different parts: a) the *RESTful APIs* to provide business logic as per the system’s functional requirements, and b) the *Service Interface* to connect to notification server, which communicates with the specific client/user to recommend some fitness activities/fitness partner at runtime.

C. Calendar management

Integrating multiple calendars to the system is very crucial and requires scalable and flexible design. This is achieved by employing the adapter design pattern. As different calendar service providers are having different data format and APIs, adapters can be created for each service provider. Such design is highly scalable as support for new service provider can be added easily without modifying the whole modular code base.

D. Data Pre-processing and classification

This module focuses on the preprocessing and classification of users’ activities and provides information about the ETL process carried out to store the classified and processed data. This process consists of three main parts: data collection, data pre-processing and storage, and data classification.

Data Collection. The accelerometer data is collected from user’s mobile device. It provides x, y and z co-ordinates with respect to the surface on which the user is performing the fitness activities along with timestamp.

Data pre-processing and storage. The activity data needs to be first preprocessed to identify features within a specific time window that will be used as input to the classification process. The time windows are parameters of the system and can be different for each user and each activity. Figure 2 details the process of defining these windows for a particular interval.

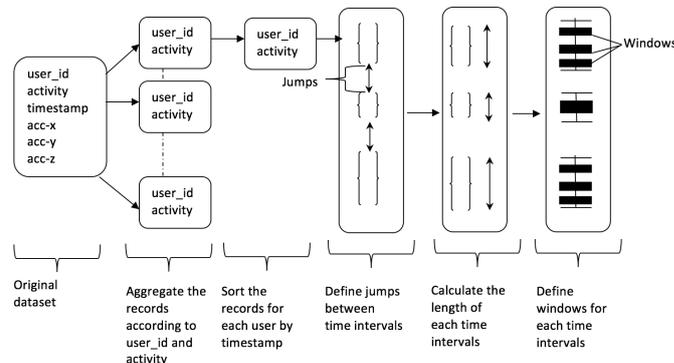


Figure 2. Data processing architecture

In the first step, the records in the original data store are grouped by user id and activity. After grouping is done, the records are sorted by timestamp in ascending order. The next step is to identify the *jumps* that are durations when no data is collected. The jump time interval is a parameter of the framework that needs to be determined experimentally. In our framework, this jump time interval is set to 5 minutes. Finally, we determine the length of each interval (*time window*) for which we want to identify the user activity. We experimented with intervals of 15, 10 and 5 seconds.

The amount data that needs to be collected is very big as it is generated every 50 milliseconds. We use Apache Kafka to stream the data in real time, while Cassandra is the best selection for storing time-series data as it has the capacity to handle real time requests. The data stored in Cassandra is ingested by feature calculation mechanism and then used by the application’s classifier algorithm to classify the user’s activity in near real time. The decoupling between data generation and classifier via Kafka would provide high scalability with increasing users.

Data classification. This is the core component of this module. In the current implementation, the user activities are classified in one of the following categories: *walking, jogging, running,*

climbing stairs, descending stairs and cycling. The classifier needs to be designed to classify each category with high accuracy, since some activities present very similar characteristics (for example, running and jogging). In order to identify user activity for a particular time interval we follow an approach inspired by the work of [11, 12], and identify the following features:

- *Average acceleration* (calculated for each axis).
- *Standard deviation* (calculated for each axis).
- *Average absolute difference* (the average of the difference between the value of each input sample records and mean of the total input sample records, calculated for each axis).
- *Average resultant acceleration* (the average of the square root of sum the squares of values of each axis).
- *Time between peaks* (the time in milliseconds between the peaks in the sine wave for each axis).

The features are calculated for a particular window size and then the user's activity is classified based on the features calculated for each window. The feature "time between peaks" is useful to find repetitive patterns such as walking and jogging. We have experimentally evaluated various algorithms and present the most interesting findings in Section V.

E. Data Stores

The framework includes multiple types of data stores on the basis of the type of data. *User activity data store* stores user's physical activity data. This data can be used in machine learning to train the model for better prediction and recommendations. *User profile data store* stores data related to the user profile, their connections, preferences and settings. *Data Warehouse* will take data from a data cleansing/ETL module will convert raw data to a standardized schema that will be useful for data analytics.

F. Visualization

This module is responsible for generating graphs or trends that would provide valuable insights to users. The main components of this module are described here. Due to huge amount of user activity data, we use Kafka to stream activity data from the application database for ETL processing Spark job since it promises high throughput [13]. To provide near real time analytics, we decided to use Apache Spark that provides a streaming library that can take input from Kafka, process it and output to any data source. One of the important tasks in the ETL is mapping of activity with goal and session based on its start time and end time that is performed in Spark. The output of Spark job will be store in a warehouse. Data Visualization module fetches data from the data warehouse and displays them in various graphic visualizations.

G. Fitness Session Recommendation Engine

This component uses user profile and activity data to recommend fitness sessions to a particular user on the basis of different parameters. This module takes following attributes as an input:

- *Type of Activity*: Physical activity like jogging, running etc.
- *Session*: Time period when the user performs a specific fitness activity.
- *Calendar Event*: A time period on a specific day.
- *Calendar*: Calendar for scheduled fitness activities.

- *Availability*: Time periods when a specific user has no calendar event.
- *Blocked Time Slots*: When user doesn't want fitness session.
- *Goal*: Time period, during which a user targets to perform certain activities/sessions to achieve a specific goal given in terms of "calories burnt" or "distance walked" etc.
- *User Strength*: A user's capacity to perform specific activity for a specific time period. Calculated using BMI (Body Mass Index).

For session recommendation, system performs two activities: 1. Generate a session for a user based on his strength, goal and activity preference. 2. Generate list of calendar events for the sessions generated in step 1 based on user's availability. Below is the outline of how above two functions would work

Session generation. Time duration of daily fitness Session can be calculated from given *Goal* (calories to burn and time duration to accomplish the target), user's fitness *Activity preference* and user's BMI (Body Mass Index).

Recommendation process. Considering user's *Availability* and *Blocked Time Slots* by syncing all user's calendar events, the framework generates a personalized list of recommended fitness Sessions as a Calendar Event in the fitness Calendar, as follows:

1. Upon sign up, the user enters all the personal information (e.g. Calories, height, weight, activity preference etc.)
2. After registration, PRO-Fit retrieves all the contacts and calendars event from the user's device into the application.
3. PRO-Fit inputs the user profile in a hybrid recommender system and generates a list of recommended fitness sessions. Recommendations are based on user's preferences (using content-based recommendations), as well as these of similar users (using item-based collaborative filtering).
4. User can reschedule session anytime. PRO-Fit will smartly scan for all the other available slots in the calendar and schedule that session.
5. A periodic sync of calendars, as well as updates on the user profile (including new activity recommendations) will be supported by the system.

When a session is just about to start, the app will send a push notification on the user's device before a certain amount of time to give a gentle reminder.

H. Social Ranking Recommendation Engine

This module integrates with the user's social network, and ranks and recommends users or friends as "fitness buddies" to the user. Fitness challenges, goal tracking and regularity in fitness activities are decision factors to decide relative ranking. This module will be developed as part of future work.

V. EXPERIMENTAL EVALUATION

Identifying user activities based on accelerometer data can be defined as a classification problem. We experimentally evaluated various classification algorithms to identify the most appropriate ones for this task. Since there is no single platform that provides all libraries, we the *Decision Tree*, *Random Forest*, and *Multinomial Logistic Regression* implementations of Spark's MLlib, the *Extremely Randomized Trees*, *AdaBoost*, *K-NN*, *Naïve Bayes* and *SVM (multi-class)* implementations of Python's Scikit-learn, and also implemented the *Gradient*

Boosted Trees algorithm, mainly to analyze the effect of this ensemble method on creating accurate model.

Raw input labeled data Statistics	
Number of inputs records	2,980,764
Class Distribution	
Walking	1,255,923 (42.1%)
Jogging	438,871 (14.7%)
Stairs	57,425 (1.9%)
Sitting	663,706 (22.3%)
Standing	288,873 (9.7%)
Lying Down	275,967 (9.3%)

Table 1 Raw input labeled data statistics

For our experiments, we used the dataset collected by Kwapisz et al. [11], publicly available through the WISDM lab [14], including activity data collected from android devices. The data was collected at an interval of 50ms, which means it contains 20 samples per second. It contains user information along with their tri-axial axis information and timestamp when it was collected. Each axis represents the relative movement of the user with respect to the surface where the user activity was performed. The statistics of the data set are included in Table 1.

We first calculated the feature set as described in section IV. Once the feature set is calculated, these features are used as input to the classifier algorithms. The experiments were carried out for sampling rate of 300, 200 and 100 records of test data to measure the accuracy for each algorithm. We used two validation methodologies, namely holdout set validation (with a 60/40 split for training/test data respectively), and 10-fold cross validation. Figures 3 and 4 show the comparison of different algorithms for different window sizes (we omit the rest due to space constraints). We observe that as we decrease the window size the number of samples increases and we get and better accuracy for our classifiers. In terms of accuracy, multiple algorithms perform well, with the decision tree-based ones consistently giving better results for all window sizes.

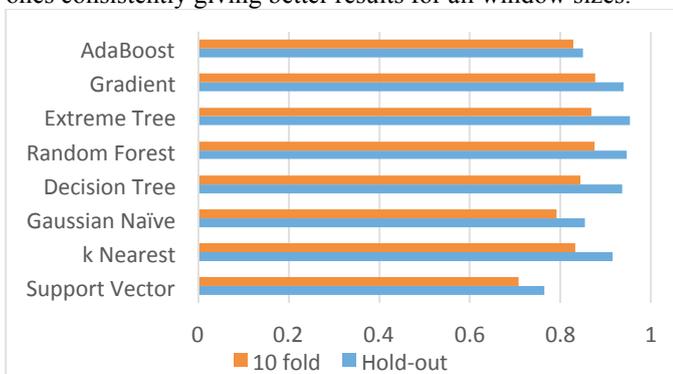


Figure 3. Comparison of all algorithms - window size of 10 seconds

VI. CONCLUSIONS

In this paper we presented PRO-Fit, a novel framework for personalized fitness recommendations. Apart from presenting details on the system architecture and module design, we experimentally evaluated several machine learning algorithms concluding that decision tree-based algorithms are the best choice for classifying activity data. In the next phase of this project we plan to focus on incorporating the social network of the user, by designing recommendation algorithms to suggest “fitness buddies” based on similar users and proximity.

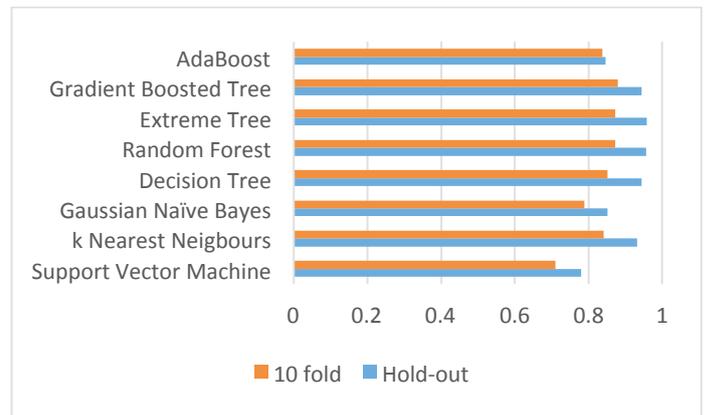


Figure 4. Comparison of all algorithms - window size of 5 seconds

REFERENCES

- [1] “MobiHealthNews: Half of mobile health app users are using fitness apps” [Online]. Available at <http://mobihealthnews.com/30199/half-of-mobile-health-app-users-are-using-fitness-apps/>
- [2] Jin-Hyuk Hong; Ramos, J.; Dey, A.K., "Toward Personalized Activity Recognition Systems With a Semipopulation Approach," in IEEE Trans. on Human-Machine Systems, 46(1), pp.101-112, Feb. 2016
- [3] Bajpai, A.; Jilla, V.; Tiwari, V.N.; Venkatesan, S.M.; Narayanan, R., "Quantifiable fitness tracking using wearable devices," in 37th IEEE Intl. Conf. of the Engineering in Medicine and Biology Society (EMBC), Aug. 2015
- [4] Ayu, M.A.; Mantoro, T.; Matin, A.F.A.; Basamh, S.S.O., "Recognizing user activity based on accelerometer data from a mobile phone," in 2011 IEEE Symposium on Computers & Informatics (ISCI), March 2011
- [5] Anjum, A.; Ilyas, M.U., "Activity recognition using smartphone sensors," in 2013 IEEE Consumer Communications and Networking Conference (CCNC), Jan. 2013
- [6] Ghose, S.; Barua, J.J., "A systematic approach with data mining for analyzing physical activity for an activity recognition system," in Intl. Conf. on Advances in Electrical Engineering (ICAEE), Dec. 2013
- [7] Guiry, J.J.; Van de Ven, P.; Nelson, J., "Classification techniques for smartphone based activity detection," in 11th IEEE Intl. Conf. on Cybernetic Intelligent Systems (CIS), Aug. 2012
- [8] Gupta, N.; Jilla, S., "Digital Fitness Connector: Smart Wearable System," in 1st Intl. Conf. on Informatics and Computational Intelligence (ICI), Dec. 2011
- [9] Uddin, M.T.; Uddiny, M.A., "Human activity recognition from wearable sensors using extremely randomized trees," in 2015 Intl. Conf. on Electrical Engineering and Information Communication Technology (ICEEICT), May 2015
- [10] Yingying Li; Yimin Zhang, "Application of data mining techniques in sports training," in 5th Intl. Conf. on Biomedical Engineering and Informatics (BMEI), Oct. 2012
- [11] J. R. Kwapisz, G. M. Weiss, and S. A. Moore, “Activity recognition using cell phone accelerometers,” ACM SIGKDD Explorations Newsletter, pp. 74–74, 2011.
- [12] M.-K. Suh, A. Nahapetian, J. Woodbridge, M. Rofouei, and M. Sarrafzadeh, “Machine Learning-Based Adaptive Wireless Interval Training Guidance System,” Mobile Networks and Applications Mobile Netw Appl, pp. 163–177, 2011
- [13] J. Kreps, “Benchmarking Apache Kafka: 2 Million Writes Per Second (On Three Cheap Machines),” [Online]. Available at: <https://engineering.linkedin.com/kafka/benchmarking-apache-kafka-2-million-writes-second-three-cheap-machines>. [Accessed: Jul-2015].
- [14] Lockhart, J., Weiss, G., Xue, J., Gallagher, S., Grosner, A., & Pulickal, T. (n.d.). “Design considerations for the WISDM smart phone-based sensor mining architecture.” In 5th Intl. Workshop on Knowledge Discovery from Sensor Data - SensorKDD '11

A Machine Learning Approach for Developing Test Oracles for Testing Scientific Software

Junhua Ding

Department of Computer Science
East Carolina University
Greenville, NC, USA
dingj@ecu.edu

Dongmei Zhang

School of Computer Science
China University of Geosciences
Wuhan, China
jjilee@163.com

Abstract—Absence of test oracles is the grand challenge for testing complex scientific software. Metamorphic testing is the novel technique for developing test oracles on metamorphic relations. Although it is easy to find metamorphic relations based on general guidelines and domain knowledge, the ones that can adequately test the software are difficult to be developed. This paper introduces a machine learning approach for iteratively developing metamorphic relations. The approach develops initial metamorphic relations and tests first, and then the relations and tests are refined through mining the initial test execution and evaluation results with machine learning algorithms. The approach and its effectiveness are illustrated through testing an open source discrete dipole approximation program.

Keywords—metamorphic testing, metamorphic relation, test oracle, scientific software, machine learning.

I. INTRODUCTION

Scientific software is the software that includes computational components for supporting scientific investigation and decision making [8]. The examples of scientific software include simulation software of nuclear reactions, software for predicting and tracking hurricanes, and software for analyzing medical images. Testing scientific software faces many challenges. Many of them are “non-testable” due to the absence of test oracles [2]. Oracle problems are key to solve for adequately testing scientific software [8]. Metamorphic testing [2] as a novel software testing technique is a promising approach for solving oracle problems. It creates tests according to metamorphic relationship (MR) and verifies the predictable relations among the actual outputs of the related tests. It was first proposed by Chen [2] for addressing oracle problems, and it has been applied to several domains such as bioinformatics systems, machine learning systems, compilers, and scientific software [16]. However, the application of metamorphic testing to large scale of scientific software is rare because of the difficulty of the identification of MRs.

Existing testing approaches normally check the correctness of each individual execution, but not the relation among outputs of multiple executions. The success of metamorphic testing tells us that checking the relation among multiple test outputs is necessary for testing scientific software that is hard to find a test oracle. However, the quality of metamorphic testing is highly depended on MRs. Due to the grand challenge to develop

strong MRs, many MRs used for testing a complex software system are relatively too weak to ensure the testing quality. It is important to define a set of criteria for evaluating the adequacy of MRs. Then the test execution and evaluation results shall be used for guiding the generation of adequate MRs [4]. The approach introduced in this paper includes a framework for the development of MRs and tests, and a strategy for refining the relations and tests through mining the test execution and evaluation results with machine learning algorithms. The test evaluation consists of test coverage evaluation and mutation testing. The mutation testing includes two steps: the first one is applied to the mutated metamorphic relations, and the second one is applied to the mutated program. The proposed approach is an enhancement of metamorphic testing in the development of test oracles for testing “non-testable” scientific software. A case study of testing an open source discrete dipole approximation (DDA) program called ADDA[19][21] is conducted to explain the approach and to demonstrate its effectiveness.

ADDA is a fairly complex scientific software system with many characteristics that cause the testing challenges that were described by Kanewala and Bieman [8]. The most challenge issue for testing ADDA is the absence of test oracles since a test input of ADDA may include thousands of parameters and it is impossible to know the correctness of the output for an arbitrary input. ADDA as open source software, its source code, manual, and other documents are available online. Anyone who is interested in the technique discussed in this paper has the opportunity to reproduce the experiment. The experience from testing ADDA can be easily shared in the software testing community. The iterative metamorphic testing for developing test oracles with machine learning for adequately testing ADDA can be easily extended for testing other scientific software.

The rest of this paper is organized as follows: Section 2 describes the general idea of the proposed approach. Section 3 discusses iteratively developing test oracles for testing ADDA. Section 4 describes related work, and section 5 concludes this paper.

II. ITERATIVE METAMORPHIC TESTING

In iterative metamorphic testing, MRs serve as test oracles, and test coverage evaluation and mutation testing are used for evaluating the test adequacy and iteratively producing adequate MRs for testing the Software Under Test (SUT).

A. The Approach

The iterative metamorphic testing consists of three major steps: development of initial MRs and tests, test evaluation, and refinement of MRs. (1) *Development of initial MRs and tests.* Based on domain knowledge of the SUT and general framework of metamorphic testing [13], one develops a set of MRs. Based on general test generation strategies such as combinatorial technique, category-based or random approach to produce tests for each MR. (2) *Test evaluation.* As soon as the SUT passes all tests, tests created for mutations of MRs are used for checking the quality of MRs. A mutation test is a set of valid tests whose outputs violate an MR. The purpose of testing SUT with mutation tests is to ensure the relation can differentiate positive tests from negative ones. The effectiveness of the test is then evaluated with selected test coverage criteria and mutation testing. A mutant should be killed by an MR or weakly killed by a test. A mutant is weakly killed when the output of the mutated program is different to the predicted one based on refined MRs. (3). *Refinement of MRs.* It is the process for creating test oracles, which can be developed through refining current MRs or defining new MRs. Machine learning algorithms could be used for processing existing test execution data and test evaluation results to find patterns for severing MRs.

B. A Running Example

This section describes the proposed approach with a running example: testing a program that is used for calculating the contrast of an image, which is 101 pixels * 101 pixels in gray scale with 8-bit resolution like those shown in Fig. 1. We define the contrast based on the average intensities of a concentric ring area and a circle area in an image. The intensity of each pixel is the pixel value such as 125 or 24 of an 8-bit resolution image. The contrast C of an image is defined as follows:

$$C = \frac{\bar{R}_c - \bar{R}_p}{\bar{R}_c + \bar{R}_p} \quad (1)$$

where \bar{R}_c is the image intensity averaged over a circle area of 18 pixels diameter centered at the origin and \bar{R}_p is the intensity averaged over a concentric ring area in the region with 30 and 66 pixels as the inner and outer diameters, respectively. There is no easy way to decide the correctness of the calculated contrast of an image, which is exactly the purpose of building the program. Iterative metamorphic testing can be used for testing the program.

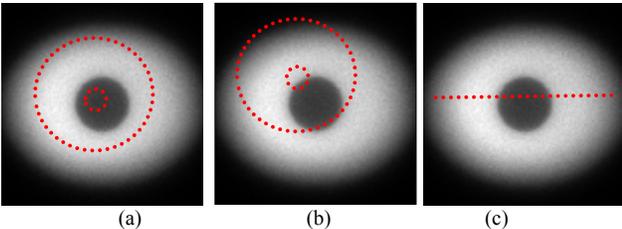


Figure 1: (a) The test outputs (the red dots in the ring) form a closed circle; (b) the test outputs (the red dots within the ring) form a half circle within the concentric ring, which violates circle relation MR1. (c) The tests cannot detect the error in the program.

1) Developing Initial MRs and Tests.

Based on formula (1), it is easy to define an MR like *increase/decrease* as follows:

MR1 (Increase/Decrease): *Given an image, increase the intensity of any pixel in the circle area will increase the contrast of the image, and increase the intensity of any pixel in the concentric ring area will decrease the contrast of the image.*

Although MR1 is useful for testing the program, it isn't good enough. For example, MR1 cannot verify whether a pixel is correctly calculated for the concentric ring or the circle area. If the concentric ring or the circle area is incorrectly implemented, it is difficult to detect the defect using the tests based on MR1 since only very few special tests will violate the MR. For example, if the program calculates the circle area using diameter 17 instead of 18, then only tests that are located on the boundary of diameter 18 may detect the error. If the origin position of the circle is set to position $\langle 50, 51 \rangle$ instead of $\langle 50, 50 \rangle$, it is almost impossible to find a test to detect the error without checking the relation among outputs of multiple tests. Therefore, it is necessary to add new MRs for checking the implementation of the concentric ring area and the circle area in the program, the relations for checking the concentric ring area and the circle area are used for guiding the selection of tests and checking the testing results. We know the center of the image is at position at $\langle 50, 50 \rangle$, and then the test oracles for deciding whether a pixel $\langle x, y \rangle$ is within the circle area or within the concentric ring area are defined as follows:

Test oracle 1 (a pixel is within a circle): *pixel $\langle x, y \rangle$ is within the circle if $\sqrt{(x - 50)^2 + (y - 50)^2} < 18/2$.*

Test oracle 2 (a pixel is within the concentric ring): *pixel $\langle x, y \rangle$ is within the concentric ring area if $(30/2)^2 < \sqrt{(x - 50)^2 + (y - 50)^2} < 66/2$.*

A test for checking the implementation of the areas is a pixel $\langle x, y \rangle$ in an image, and it is marked with a special color when the pixel is counted for an area during testing. Each MR1 test is a candidate for creating new tests based on new MRs for metamorphic testing. The relation among the outputs, which are colored pixels, can be visually verified. MR2 is defined as follows:

MR2 (circle): *Given a set of test inputs $t_1 = \{\langle x_i, y_i \rangle | \sqrt{(x_i - 50)^2 + (y_i - 50)^2} = 32\}$, the outputs shall form a circle that is within the concentric ring area and has the same origin as the ring. Given a set of test inputs $t_2 = \{\langle x_i, y_i \rangle | \sqrt{(x_i - 50)^2 + (y_i - 50)^2} = 8\}$, the outputs shall form a circle that is within the circle area and has the same origin as the circle.*

t_1 and t_2 form a circle in the circle area and a circle in the concentric ring area, respectively, and their outputs should form a colored circle in the circle area and one in the concentric ring area.

2) Test Evaluation

As soon as the program passes all tests, it is further tested with mutation tests. Testing with mutation tests is a type of negative testing to determine the response of the

system with unexpected test inputs. The purpose of testing with mutation tests in this research is to check the quality of MRs and mitigate the problem that a relation is so weak that can be satisfied by any tests. A mutation test is a set of valid tests whose outputs don't satisfy an MR. A mutation test t'_1 of MR2 is defined as follows:

$$\text{Mutation test } t'_1: \{ \langle x_i, y_i \rangle | ((x_i - 50)/40)^2 + ((y_i - 50)/20)^2 = 1 \}$$

The outputs of test set t'_1 don't satisfy relation MR2 since they form an oval instead of a circle, which shows relation *circle* is a carefully selected relation and only satisfied by a carefully selected test sets. Using the same idea, one can create a mutation test for MR1. The test includes pixels outside of the areas, and changing of the intensity of the pixels would not affect the contrast.

Test adequacy is evaluated through measuring selected test coverage criteria such as function coverage, condition coverage and mutations [7]. It is infeasible to kill all mutants, and it is even more challenge to kill mutants when testing "non-testable" programs. Many mutants such as one systematically shifts the real values cannot be killed by simple MRs, which is also a reason of checking MRs with mutation tests. If the correctness of an individual output can be checked, then regular mutation testing [7] can be used for testing the program. Otherwise, we expect each mutant will be at least weakly killed. A good MR needs kill some mutants. If a mutant was not killed or weakly killed by any test, then new tests or new MRs should be developed until the mutant is killed or a conclusion of the infeasibility of killing the mutant is made. If all mutants are killed or weakly killed, it is necessary to check whether these mutants are killed uniformly by the MRs.

3) Refining MRs

When a pixel $\langle x, y \rangle$ in test t_1 or t_2 is tested, the program marks the pixel with a different color. As soon as all pixels are tested, it is easy to observe whether the outputs of t_1 or t_2 form a circle within the expected area. For example, if the concentric ring is implemented shifted from the center origin $\langle 50, 50 \rangle$ to $\langle 55, 50 \rangle$, then we can find some colored pixels are shifted from the center of the concentric ring area, and the colored pixels form an incomplete circle within the concentric ring area, as shown in Fig. 1 (b). If the test was created without considering the relation, such as those shown in Fig. 1 (c), the shifting error would not be detected since each individual output is correct. For the same reason, relation *circle* is also useful to detect the problem in test oracles. For example, if test oracle 2 incorrectly uses position $\langle 55, 50 \rangle$ as the origin of the image, the problem can be easily detected by either test inputs t_1 or t_2 .

In this research, the initial relations are revised and refined during test process and in turn the updated MRs are used for producing more tests. Let's use MR *increase* as an example to explain the refinement process. We select one image and change the intensity of some pixels in the concentric ring or circle areas of the image to create a new image. In the beginning, no one knows the exact difference of the contrast between the original image and the modified one except relation MR1 of the contrasts between the two images. But as soon as the program calculates the exact numbers of pixels and their intensities in the concentric

ring and circle areas, it is easy to calculate the exact difference between the contrasts of two images that have different intensity for exact one pixel so that MR *increase* is refined from a general relation to a precise one. For example, if a pixel's intensity in the circle area is increased from 64 to 128, and we know there are n pixels with total intensity I_c in the circle area based on previous execution, one can use new $\bar{R}_c = (I_c + 64)/n$ to calculate the exact contrast of the new image using formula 1. The MR *increase* is refined with exact values. The refined relations are used for creating more tests for testing the program, and each individual output can be precisely verified. In order to develop MRs for complex scientific software, machine learning of the test execution data and evaluation results should be used.

III. TESTING ADDA USING METAMORPHIC TESTING

In this section, we discuss testing ADDA using the iterative metamorphic testing, especially on refinement and development of MRs using machine learning approaches.

A. Testing ADDA

DDA is a method to simulate light scattering from particles through calculating scattering and absorption of electromagnetic waves by particles of arbitrary geometry [19]. The particle as a dielectric scatterer are divided into many small volumes called dipoles. The dipoles make up a small cubic lattices of spacing within a fraction of wavelength of the incident of light and they are each exposed to the incident field and the field due to all other dipoles. The interactions of dipoles are approximated based on equations of the electric field [19]. DDA has been widely used for light scattering simulations [19]. ADDA is an open source implementation of DDA written in C99 with routines in Fortran and C++ [21]. The general input parameters of ADDA define the optical and geometry properties of a scatterer/particle including the shape, size, dipoles, refractive index of each dipole, orientation of the scatterer, definition of incident beam, and many others. ADDA produce different outputs for different applications such as Muller matrix at different scattering angles used for producing diffraction images. It is unknown the correctness of an ADDA simulation with an arbitrary heterogeneous scatterer with an arbitrary shape in advance. The authors of ADDA have conducted extensive testing of ADDA for special cases, but more general tests of ADDA is still necessary. Here we discussed how to test ADDA using iterative metamorphic testing, particularly on developing MR using a machine learning approach.

B. Development of Initial MRs and Tests

ADDA has been extensively tested with special cases [14][21], which serve as the initial tests for creating tests using MRs. Since an output of ADDA may include thousands items and its input may include lots of parameters, it is very difficult to find an MR directly on its inputs and outputs. We define MRs on the textual property of the the diffraction image generated from an ADDA output. The textual patterns of diffraction images can be visually observed or quantitatively characterized with Grey Level Co-occurrence Matrix (GLCM) features [6]. GLCM

calculates computable textural features based on grey-tone spatial dependencies. It defines how often different combinations of gray level pixels occur in an image for a given distance d in a particular angle θ [6]. We define initial MRs based on the shape, size, orientation, and refractive index of a scatterer. Each MR checks the correlation between a parameter and the textual pattern of the ADDA calculated diffraction image of a scatterer.

MR3 (Difference): *When the size, shape, orientation, refractive index value of a scatterer is changed, its textual pattern of the diffraction image is changed. Only one parameter is changed at each time, and the change of the orientation doesn't affect the textual pattern of a sphere scatterer.*

Since the ADDA simulation results of sphere scatterers have been compared to the results calculated from Mie theory with many different configurations [14]. We assume the calculation of ADDA on sphere scatterers is correct. For testing each MR, we first compare the result to a sphere scatterer result. Fig. 2 shows the comparison of the textual patterns of three scatterers in different shapes, and the result satisfies MR3.

We use combinatorial technique to create tests to cover more scenarios. For example, the four input parameters are the *scatterer size, shape, refractive index, and orientation*. Then select the possible values for each parameter guided by techniques such as category based technique, random, or boundary values. The base tests can be created using combinatorial techniques such as pairwise, and then the MRs are used for producing tests based on the base tests. For example, the possible values of size are $\{3\mu\text{m}, 5\mu\text{m}, \dots, 16\mu\text{m}\}$ for the ADDA study, shapes are $\{\text{sphere, ellipsoid, bi-sphere, prism, egg, cylinder, capsule, box, coated, cell1, cell2, \dots}\}$, orientations are $\{<0, 0, 0>, <10, 90, 0>, <270, 0, 0>, \dots\}$, and refractive index values are $\{1.0, \dots 1.5\}$. Using pairwise, one can create many base tests, and then select the valid tests as the first tests to create MR tests for MR3. Fig. 3 shows a comparison of the textual patterns of diffraction images of a cell with different refractive index values in nuclear. The result satisfies MR3.

Although MR3 can offer some preliminary testing for ADDA for heterogeneous scatterers including real cells, the relation is still too weak to adequately test ADDA. MR3 should be refined and probably new MRs should be created for testing the program.

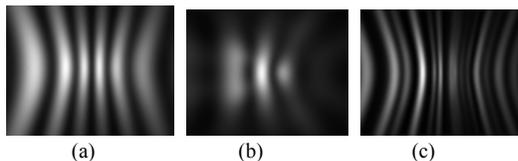


Figure 2. The comparison of the textual patterns of diffraction images of scatterers in different shapes (a) sphere, (b) ellipsoid, and (c) bi-sphere.



Figure 3. The comparison of the textual patterns of diffraction images of a cell with different refractive index values in nuclear.

C. Evaluation of MRs and Tests

The initial tests created using MR3 covered 100% statements and close to 100% of conditions. Partial mutation testing was conducted to check the effectiveness of the metamorphic testing. The mutation testing was applied only to one module of ADDA program. Several mutants were instrumented to the code manually and their results were manually inspected.

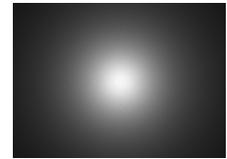


Figure 4. An error diffraction image.

Mutation testing of “non-testable” programs like ADDA is difficult since many MRs are not sensitive to mutants. In this case, one can check the consistency between the outputs of the mutated program and the original one. A mutant is killed if it causes a violation of any MR, and a mutant is weakly killed if the output of a test of the mutated program is different to the original one. Absolute Value Insertion (ABS) and Relational Operator Replacement (ROR) were used for creating mutants. Among total all 20 mutants (10 ABS mutants, and 10 ROR mutants) we created for testing ADDA, 17 of them were killed by crashing of the program or exception handling. The other 3 mutants were killed by the MRs since the outputs didn't generate any textual pattern as shown in Fig. 4. We found mutants are easily killed by simple tests in scientific software probably due to the complexity of the software. A slight change of the program can cause a catastrophic error in the calculation. The mutant that instrumented to ADDA program produced the diffraction image shown in Fig. 4 is easily to be killed since the two different scatterers produced the diffraction images with the same textual patterns. The results show that MR3 can test ADDA in some degree.

However, it is very difficult to create a mutation test for MR3 that would produce two diffraction images with the same textual patterns by changing the input parameters. MR3 is so weak that many wrong results can satisfy it.

D. Refinement of MRs

According to the results discussed in Section III. B, C, it is easy to see the intuitive relation between the textual pattern of a diffraction image and its 3D morphology. We need refine the relation for understanding how the change of 3D morphology parameters including the shape, size, refractive index and orientation are precisely related to the textual pattern. We investigate the problem via an experimental study. First, we took many diffraction images of different types of cells using a diffraction image based flow cytometer called p-DIFC. These images are called measured diffraction images to compare the calculated diffraction images produced from ADDA. Second, process these measured images for GLCM, and check how the GLCM features are related to cell types. Select optimal GLCM features for cell classification, build a feature vector including labeled cell type for each image, and construct a feature vector matrix with the images that are belonged to the same type of cells for training a SVM. Third, use the trained SVM to classify diffraction images for cell types based GLCM feature values. If the cells can be successfully

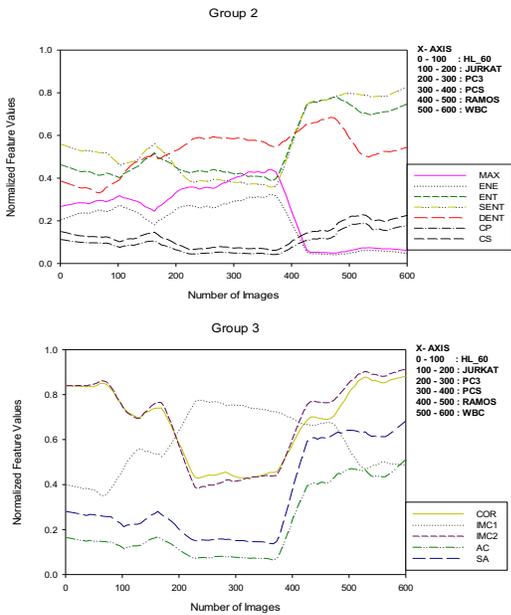


Figure 5. GLCM features of 6 types of diffraction images

classified by the GLCM features, then we need test whether the calculated diffraction images can be also used for classifying the cell type. If it does, the cell classification based on diffraction images will serve as a test oracle for testing ADDA.

We took 100 diffraction images for each type of cells for total 6 different types of cells using p-DIFC. Fig. 5 is the experimental result of selected GLCM features among 600 cells. From the two diagrams, it is not difficult to see that the correlation between some GLCM features and the cell types. However, the cell classification cannot be completed just based on one feature, instead multiple features have to be used. In this study, feature selected was conducted, and 8 GLCM features were selected for the SVM based classification [17]. Fig. 5 should be able to serve as a test oracle for testing ADDA since we expect the ADDA calculated diffraction images of the same type of cells in Fig. 5 should also have the same feature patterns as those shown in Fig. 5. However, the real reflective index value of each cell organelle is unknown but a guessed one, the feature patterns between the measured images and calculated images should be similar but not identical, and the precise relation between them cannot be defined. Fig. 5 can serve as a reference for testing ADDA, but it is not enough.

The SVM based classifier is built on Weka with SVM library LIBSVM. Stratified 10-Fold Cross Validation (10FCV) was used for checking the accuracy of the SVM classification. In this study, specifically 90 images per each type of cells were used for training the SVM and remaining 10 images were used for testing the classifier. We experimented the classification using different sets of GLCM features and different distance and grey level for calculating GLCM. The best performance of the SVM classification is configured with 8 selected GLCM features, and the GLCM calculation was configured with distance as 2 and grey level as 64. The average accuracy of the classification of the 6 types of cells with total 600

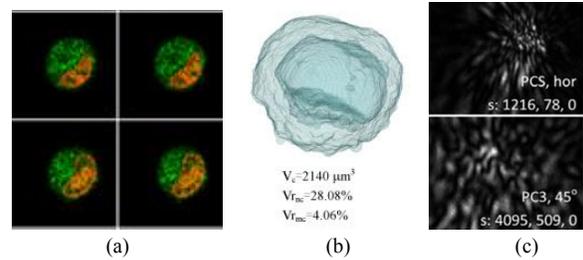


Figure 6. Confocal images (a), 3D structure (b) and diffraction images (c).

diffraction images is 91.16% [17]. Based on the experimental result, we expected the calculated diffraction images should have the same property as the measured images, which can classify cell types based on the GLCM features. However, we cannot use the SVM classifier trained with the measured diffraction images for classifying the calculated images, which has to be trained using the calculated diffraction images.

In order to calculate a diffraction image of a cell using ADDA, we need to construct the 3D structure of the cell and then assign the refractive index values for each voxel of the cell. A stack of confocal image sections are taken using a confocal microscope, and then each image section is processed to segment the cell components such as nuclear and mitochondria. The 3D structure of the cell is built based on the processed image sections, and a refractive index value is assigned to each voxel in the structure. The 3D structure then is imported to ADDA for the simulation, and a diffraction image is generated from the simulation result. Since orientation is one of the input parameters to ADDA, but cell type doesn't have any relation to the orientation. Therefore, we use ADDA to calculate diffraction images in many different orientations for each cell. Fig. 6 shows 4 of total 48 confocal image sections of a cell, its 3D structure and calculated diffraction images in two different orientations.

We took confocal image for 30 cells each type of cells, and 4 types of cells were used in the experiment. The 3D structure of each cell was simulated for 25 different orientations in ADDA. Therefore, we calculated 750 diffraction images for each type of cells. The images are processed for GLCM features with distance 2 and grey level 64, and a feature vector matrix including the feature values and labeled cell type is built for the same type of images. Finally, the feature vector matrix is used for training the SVM and 10FCV is used for checking the accuracy of the classification. Our preliminary result has shown the cells can be classified with the accuracy as high as the measured diffraction images. However, we haven't completed the experiment for all 4 type cells. Based on the experimental result, a new MR is developed:

MR4 (Cell classification): *A calculated diffraction image of a cell can be correctly classified into a classification by a SVM classifier. If two calculated diffraction images produced from two different types of cells, an SVM classifier can classify each into the correct classification.*

The new MR built based SVM classifier can test the program using existing data. The data size used for this study is very limited, it is still not clear whether the approach can be extended to all different types of cells or even different scatterers. However, the approach should be very useful in general for developing test oracles for testing scientific software that is absent of test oracles.

IV. RELATED WORK

One of the greatest challenges for testing scientific software is due to the oracle problem [8][1]. Metamorphic testing is the technique for addressing the oracle problem though developing them with MRs [2][16]. It has been applied to several domains such as bioinformatics systems, machine learning systems, and online service systems [10][11][20]. Murphy et al. classified six types of MRs for testing machine-learning systems [15], which are useful for creating initial MRs. Xie *et al.* investigated some specific MRs that were extended from general MRs for testing machine learning applications [18]. Guderlei and Mayer proposed a statistical metamorphic testing in [5], where two or more output sequences were generated and compared according to the statistical MRs. The similar idea is used in this paper for developing test oracles based on cell classification. Metamorphic testing has been also used for testing scientific software. For example, Mayer and Guderlei developed a group of MRs for testing image process programs [13]. Chen, Fend and Tse have applied metamorphic testing for testing partial differential equations [3]. However, the program they tested were much smaller and simpler than ADDA. Knewala, Bieman and Ben-Hur recently reported a result on the development of MRs for scientific software using machine learning approach integrated with data flow and control flow information [9]. However, whether the result can be extended for testing large scientific software is unclear.

V. SUMMARY

In this paper, we introduced an iterative metamorphic testing technique for testing scientific software, where MRs are iteratively developed based on analyzing test execution and evaluation results. We illustrated the approach and its effectiveness through testing ADDA, the widely used open source scientific software. A new MR was developed through analyzing experimental data using machine learning algorithm SVM. The approach could be useful for testing similar scientific software as well as other software systems that are absent of test oracles.

ACKNOWLEDGMENT

The authors would thank Dr. Xin-Hua Hu, Eric King at East Carolina University for assistances of the experiments. This research is supported in part by grant CNS-1262933 and CNS-1560037 from the National Science Foundation.

REFERENCES

- [1] E.T. Barr, M. Harman, P. McMinn, M. Shahbaz, S. Yoo, "The Oracle Problem in Software Testing: A Survey," IEEE Trans. on Software Engineering, , vol.41(5), pp.507-525, 2015.
- [2] T. Y. Chen, S. C. Cheung, and S. Yiu, "Metamorphic testing: a new approach for generating next test cases", Tech. Rep. HKUST-CS98-01, Dept. of Computer Science, Hong Kong Univ. of Science and Technology, 1998.
- [3] T. Y. Chen, J. Feng and T. H. Tse, "Metamorphic testing of programs on partial differential equations: a case study," COMPSAC 2002. pp. 327-333.
- [4] J. Ding, T. Wu, J. Q. Lu, X. Hu, "Self-Checked Metamorphic Testing of an Image Processing Program," 4th Intl. Conf. on Security Software Integration and Reliability Improvement, Singapore, 2010.
- [5] R. Guderlei, and J. Mayer, "Statistical metamorphic testing - testing programs with random output by means of statistical hypothesis tests and metamorphic testing", in Proc. of the 7th ICQS. pp. 404-409, 2007.
- [6] R. M. Haralick, K. Shanmugan, and I. H. Dinstein, "Textural features for image classification", IEEE Trans. Syst., Man, Cybern., vol. SMC-3, pp.610 -621, 1973.
- [7] Y. Jia; M. Harman, "An Analysis and Survey of the Development of Mutation Testing," IEEE TSE, vol.37, no.5, pp.649-678, 2011.
- [8] U. Kanewala, J. M. Bieman, "Testing scientific software: A systematic literature review", Information and Software Technology, Vol. 56, Issue 10, Oct. 2014, pp. 1219-1232, 2014.
- [9] U. Kanewala, J. M. Bieman, A. Ben-Hur, "Predicting Metamorphic Relations for Testing Scientific Software: A Machine Learning Approach Using Graph Kernels", Journal of Software Testing, Verification and Reliability, Nov. 16, 2015, DOI: 10.1002/stvr.1594.
- [10] V. Le, M. Afshari, and Z. Su. "Compiler validation via equivalence modulo inputs". In Proceedings of the 35th ACM SIGPLAN Conference on PLDI '14. pp.216-226. 2014.
- [11] M. Lindvall, D. Ganesan, R. Árdal, and R. E. Wiegand. "Metamorphic model-based testing applied on NASA DAT: an experience report". Proc. of the 37th ICSE, Vol. 2. pp. 129-138. 2015.
- [12] H. Liu, F. Kuo, D. Towey, T.Y. Chen, "How Effectively Does Metamorphic Testing Alleviate the Oracle Problem?" IEEE Trans. on Software Engineering, vol.40, no.1, pp.4,22, Jan. 2014.
- [13] J. Mayer, and R. Guderlei, "An empirical study on the selection of good metamorphic relations", In proc of 30th COMPSAC, pp. 475-484, 2006.
- [14] M. Moran, "Correlating the morphological and light scattering properties of biological cells", PhD dissertation, department of physics, East Carolina University, 2013.
- [15] C. Murphy, G. Kaiser, L. Hu, and L. Wu. "Properties of machine learning applications for use in metamorphic testing". In Proc. of the 20th SEKE, pp. 867-872, 2008.
- [16] S. Segura; G. Fraser; A. Sanchez; A. Ruiz-Cortes, "A Survey on Metamorphic Testing," in IEEE Trans. on Software Engineering , vol.PP(no. 99), doi: 10.1109/TSE.2016.2532875. 2016.
- [17] S. K. Thati, J. Ding, D. Zhang, and X. Hu, "Feature Selection and Analysis of Diffraction Images", the 4th IEEE Intl. Workshop on Information Assurance, Vancouver, Canada, August 3-5, 2015.
- [18] X. Xie, J. Ho, C. Murphy, G. Kaiser, B. Xu, and T. Y. Chen, "Application of metamorphic testing to supervised classifiers". 9th Intl. QSIC '09, pp. 135 - 144, 2009.
- [19] M.A. Yurkin and A.G. Hoekstra, "User manual for the discrete dipole approximation code ADDA 1.3b4", <http://adda.googlecode.com/svn/trunk/doc/manual.pdf> (2014). Last accessed on March 25, 2016.
- [20] Z. Zhou, S. Xiang, T.Y. Chen, "Metamorphic Testing for Software Quality Assessment: A Study of Search Engines", IEEE Trans. on Software Engineering, doi:10.1109/TSE.2015.2478001, 2015.
- [21] ADDA project, <https://code.google.com/p/a-dda/>, Last accessed on March 12, 2016.

A Systematic Mapping Study on the Multi-tenant Architecture of SaaS Systems

Victor Hugo S. C. Pinto, Helder J. F. Luz, Ricardo R. Oliveira, Paulo S. L. Souza and Simone R. S. Souza
Institute of Mathematical and Computer Sciences, University of São Paulo (ICMC-USP)

São Carlos-SP, Brazil

victor.santiago@usp.br, helderfl, ricardoramos, pssouza, srocio{@icmc.usp.br}

Abstract—Background: SaaS (Software as a Service) is a services delivery model in Cloud Computing whose applications are remotely hosted by the service provider and available to customers on demand over the Internet. Multi-tenant Architecture (MTA) is an organizational pattern for SaaS that enables a single instance of an application to be hosted on the same hardware and accessed by multiple customers, so-called tenants, with the aim of lowering costs. Tenants are able to configure the system according to their particular needs. **Objective:** This research aims at the obtaining an overview of the challenges and research opportunities in MTA context for SaaS through a Systematic Mapping Study. **Results:** Eighty nine primary studies were selected for discussions on advances and opportunities for further investigations. The results showed the relevancy of MTA and pointed out the main research trends for next years in this topic.

Cloud Computing; software as a service; multi-tenant architecture; systematic mapping study.

I. INTRODUCTION

Cloud Computing has emerged from the contribution of techniques from parallel computing, distributed computing and platform virtualization technologies [1]. It provides dynamic resource allocation and has become one of the main research fields in Software Engineering. Furthermore, this technology enables cost reduction, optimization and opportunity for the creation of new business models [2]. A set of resources can be efficiently accessed on demand from anywhere and managed with a minimum possible interaction [3]. Cloud can be understood as a repository of virtualized resources (hardware, development platforms/or services) easily accessible [4]. These resources can be dynamically reconfigured to be adjusted to diversified loads, which optimizes their usage. This wide range of resources has directly contributed to the emerging of different services delivery models, as SaaS (Software as a Service), which is a software deployment model of applications remotely hosted by a service provider and available to customers on demand. It offers benefits, as improved operational efficiency and reduced costs. As an instance, Salesforce.com¹ provides an SaaS for Customer Relationship Management. Salesforce uses a subscription revenue model and charges clients per user on a monthly basis.

The cloud computing environment is different from a traditional environment in terms of hosted deployment, configu-

ration, execution and management of applications. The main difference is related to type of users, security and sharing of resources such as databases, virtual machines or network connections among customers [5]. The sharing of resources among customers through logical separation is one of the main characteristics of multi-tenant architecture (MTA) for SaaS systems.

Multi-tenancy can be referred to an organizational pattern in which a single instance of an application is hosted on the service provider, and multiple companies, so-called tenants, access the same instance [6]. MTA enables a high degree of customization of software according to the requirements of many tenants and resources required for its execution are shared and provided on demand. For the end users, the application is executed in a dedicated environment, i.e., a fault of software in use by another tenant should not affect them. Furthermore, they are able to exclusively configure the system to their specific needs. MTA provides benefits, such as (i) optimization of the use of hardware resources, (ii) costs reduction by the maintenance of applications and (iii) new opportunities for data aggregation. However, challenges as those related to security, data sharing, database, customization, validation and testing, performance and migration from conventional web applications [7][8] must be overcome.

A Systematic Mapping Study (SMS) is a proper method to map a certain topic when few evidence exists or the research topic is wide or scattered. Therefore, we have carried out an SMS on the multi-tenancy of SaaS systems following the guidelines proposed by Kitchenham [9]. Eighty nine primary studies were selected to answer two research questions from the academic perspective. The analysis of the results focuses on presenting the frequencies of publications for different research categories. As main contribution, we have provided a definition of main challenges to guide future research on the multi-tenant architecture domain.

The paper is organized as follows: Section 2 discusses the phases of the SMS; Section 3 addresses the threats to validity and Section 4 reports the conclusions and future work.

II. THE SYSTEMATIC MAPPING STUDY

The SMS was conducted considering three main phases: (i) planning, (ii) conducting and (iii) reporting. The next sections address these phases and the obtained results.

¹www.salesforce.com/sales-cloud/overview/
DOI reference number: 10.18293/SEKE2016-068

A. Planning

In this phase, the review protocol containing (i) research questions, (ii) search strategy, (iii) inclusion and exclusion criteria and (iv) data extraction process and methodology for the synthesis of the data was defined.

The main goal was the achievement of a background of difficulties related to MTA, alternatives proposed in the literature and research opportunities. Therefore, two research questions (RQ) were defined:

RQ_1 : What research topics related to MTA can be found on the current literature?

RQ_2 : What are the main research challenges and opportunities related to the development, testing and evolving of multi-tenant SaaS applications?

A search string and the electronic databases were also defined. The search string was elaborated and refined according to an initial set of key papers selected and based on citations of these papers. During the string validation these papers must always be retrieved from electronic databases (Table I). Although subjective, this control enabled the string calibration and identification of possibly relevant studies.

TABLE I. List of key papers used to calibrate the search string

Authors	Ref.
Seungseok et al.	[10]
Sengupta and Roychoudhury	[7]
Tsai et al.	[8]
Ru et al.	[11]

We defined the search string considering the following keywords: cloud, SaaS and multi-tenancy, their frequent variations and boolean operations. Figure 1 shows the search string elaborated. The following databases were considered: *ACM*, *IEEE*, *Scopus* and *Wiley Online*. Such databases cover the main conferences and journals on cloud computing.

(cloud **and** (SaaS **or** "Software as a Service") **and** (multi-tenancy **or** multi-tenant **or** tenancy **or** tenant **or** tenants))

Fig. 1: Search String.

Relevant primary studies were selected based on the following inclusion (*IC*) and exclusion criteria (*EC*). Not all inclusion criteria should be satisfied for each primary study; IC_a is the only mandatory criterion for the inclusion of papers.

IC_a : The primary study presents at least one challenge or research opportunity in the context of MTA;

IC_b : The primary study presents at least one tool, framework, process or APIs for MTA context;

IC_c : The primary study addresses at least one difficulty involving the MTA in usage and migration terms;

IC_d : The primary study presents at least one property, classification or evaluation of a solution considering the MTA;

EC_a : The study presents a challenge or a research opportunity in the MTA context. However, it is a short paper;

EC_b : The study is a Systematic Literature Review;

EC_c : The whole study is unavailable.

We have used a data extraction form to answer the review questions, presented in Table II. We have included some categories (Item 5) in order to classify the main domain of each primary study, for instance, "Customization" and "Database" are defined categories.

TABLE II. Contents of data extraction form

Attributes	
Metadata	ID, reviewer and date.
Content	Title, year, source (i.e. conference or journal) and search database.
Data extracted	1) Challenge/opportunities; 2) Tools, frameworks and APIs; 3) Difficulties in the MTA usage and adoption/migration; 4) Specifications, classification and evaluation of solutions for MTA and 5) Category of paper.

During the data extraction process, the data from primary studies were collected by three reviewers, PhD candidates in Computer Science. They were extracted by one researcher and checked by another. This SMS was performed between May and August, 2015 and the data have been documented and are available².

B. Conducting

In this phase, the primary studies were identified in the aforementioned search databases. *Scopus* returned a larger set of studies (638). *IEEE*, *ACM*, *Wiley* returned 168, 594 and 25, respectively. Figure 2 shows the distribution of papers retrieved in each search database and after applying the inclusion and exclusion criteria in the reading process. From this initial set, 135 duplicated studies were identified and removed. In the selection phase, based on the partial reading (titles and abstracts), a set of 149 papers was selected according to the inclusion and exclusion criteria; after the full reading, only 89 papers were selected. We wanted to be conservative as possible, therefore the search string has become generic to retrieve many studies from electronic databases, even if it would give us more effort in the selection process. Many papers were introduced as primary studies, but only few of them had more contributions or larger impacts.

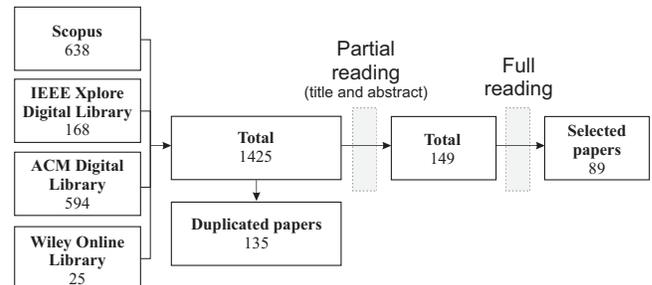


Fig. 2: Distribution of papers (conduction phase).

In order to validate the inclusion and exclusion criteria application, each primary study was scored by reviewers in

²<https://goo.gl/0K68jp>

both partial (title and abstract) and full readings. A score “0” means rejected and “1” accepted. In cases of doubt, the reviewer scored the paper with “0.5” and the other reviewers were asked about its relevance, so that a consensus could be reached through the adoption of score “0” or “1”.

The scoring process was conducted in a sequential and independent way, i.e., one reviewer read and scored each paper without interference from others. In partial reading all papers were scored and posteriorly, in full reading they were classified again by reviewers until reaching a set of relevant studies to answer the research questions.

C. Reporting

This section discusses an overview of MTA based on selected primary studies.

1) RQ₁: What research topics related to MTA can be found on the current literature?

In order to clarify the focus of the selected studies in quantitative terms, we have defined some categories according to the paper domain, as aforementioned. Figure 3 shows a mapping containing number of primary studies distributed according to publication year and category, which one paper can be classified and more than one category.

Security, testing activity and experiments may be considered important issues to quality assurance and, therefore, these issues are into quality assurance category.

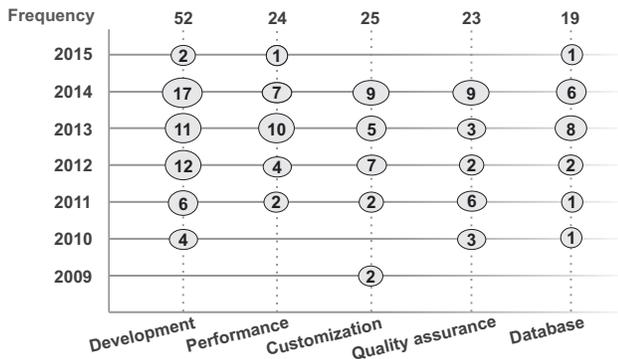


Fig. 3: Distribution of primary studies according to categories.

Figure 4 presents the disposal of the papers selected from workshops, journals and conferences. Conferences have a dominant position with 63 papers (70.8%), followed by journals with 18 papers (20.2%) and workshops, 8 papers (9%). It can indicate that workshops still have to be formed and researchers submit their results to conferences and journals with a larger scope.

The next sections discuss the main idea of the selected studies organized by the categories shown in Figure 3.

a) *Development*: Architectures, frameworks, requirements and variability management, and migration from web conventional applications to multi-tenant SaaS applications are mentioned in the current multi-tenancy literature as important issues to address in future research.

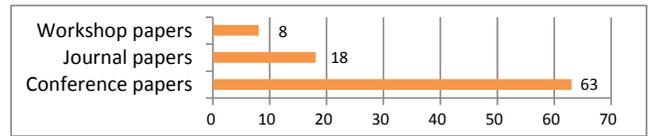


Fig. 4: Quantity of selected papers from workshops, journals and conferences.

Due to the complexity of the management and maintenance costs, SaaS providers commonly develop a single version of the application for all tenants. Truyen et al.[12] evaluated the context-oriented programming (COP) for such applications aiming to improve their development process. The main idea was to enable the customization of specific requirements for each tenant, providing positive results in terms of development effort.

Requirements management is one of the difficulties related to the supply of satisfactory applications for certain tenants. Walraven et al. [13] presented an alternative based on product lines and the co-existence of specific settings for tenants to facilitate the requirements management. They used the approach in some multi-tenant SaaS applications and as a result, the efforts to configure and compose variant applications were reduced.

Multi-tenant SaaS applications aim at providing different settings to address the demands of their tenants in functional requirement terms. However, it is natural that different requirements from the previously provided be needed by new tenants. On the one hand, new features can be developed and included in an application, so that the existing instances continue running without failure. On the other hand, costs related to efforts in the regression testing must be considered during the development of new features [7].

Kale and Borhade [14] provided a framework that covers features as portability, customization, security and scalability aiming to reduce the time spent on the software development. In the same vein, Nam and Yeom [15] proposed a framework to support different services for a large set of tenants. According to the authors, despite the benefits of MTA, few processes can support the development of multi-tenant SaaS applications.

Manduca et al.[16] described an approach for the development of multi-tenant SaaS applications with a single database from conventional web applications. It employs architectural components and design patterns and keeps the functionalities in the current programming language with no need for substantial changes. As a difficulty, the authors highlighted the platform documentation, which is often insufficient to develop this type of applications.

Table III presents the issues related to development of multi-tenant SaaS applications investigated. Due to space limitation we are showing only some papers for each issue.

b) *Performance*: MTA has introduced new challenges related to load balancing and resource allocation, including the requests on the tenant level, service level agreement, performance objectives and quality of services. Sun et al. [18]

TABLE III. Main issues about development of multi-tenant SaaS applications

	Description	Ref.
Requirements	Composition of variant multi-tenant applications following the product lines engineering	[13]
Implementing	Context-oriented programming	[12]
	Guide to implementing new tenants without impacting those already deployed	[7]
Frameworks	Process to support the portability, customization, security and scalability	[14]
	Framework to address availability, extensibility and scalability in a multi-tenant application	[17]
Migrating	Process to support the migrating of conventional web applications towards multi-tenant SaaS application with relatively less effort	[16]

proposed a suitable load balancing policy for a multi-tenant environment to provide satisfactory quality of services. On a database level, Moon et al. [19] presented a load balancer for multi-tenant databases to increase the performance and sharing of resources among tenants. Patikirikoralala et al. [20] developed an approach that uses a nonlinear replenished control to keep the performance in distinct usage levels for different tenants, depending on their priorities. It enables the detection of overload, therefore the control of tenants operations can be dynamically changed.

Krebs et al. [21] extended a web benchmark called TPC-W to include multi-tenancy and compared the cloud usage under two perspectives: (i) multi-tenancy and (ii) virtualization. Multi-tenancy shown higher efficiency than virtualization considering the throughput, number of tenants and when memory was a bottleneck.

c) Customization: A considerable number of papers have addressed applications customization. A multi-tenant SaaS application which address a large set of tenants should make possible a large number of customizations [22]. The customization of a complex application is an error-prone task, it requires high manual efforts and the users may not know the best choice in terms of customization. Thus, the authors performed a study of the possibilities available for the customization of an SaaS application, and a semi-automatic customization process was created to reduce efforts.

Ramachandran et al. [23] observe that the customization may result in high cost of readjustments. For multi-tenant systems, it involves the configuration of specific instances and management of allocated resources for the tenants. For Walraven et al. [24] customization involving variations in the core of the application is expensive for SaaS providers, introducing an additional complexity.

d) Quality assurance: Quality assurance is a promising research topic in MTA [25] that includes testing strategies, metrics and quality criteria, and alternatives related to security. For Tsai et al. [8], one of the main challenges in the testing activity of multi-tenant SaaS application is to deal with the large set of composition possibilities and interactions among

components. The authors provided a combinatorial testing approach to generate dynamic test sequences and achieve a high structural coverage. The main idea was to identify the compositions likely to result in failures by an algorithm. When a new component is composed in a certain application instance becomes available, the algorithm reveals defects in the interactions among components.

The complexity of the cloud computing model and lack of standardization become the security a critical issue for cloud providers and customers. According to Wood et al. [26], multi-tenancy directly impacts on the applications development and the way they are provided. Almorsy et al. [27] created a framework to improve collaboration between service providers and consumers and manage the security of cloud platform and its hosted services.

Table IV presents the main issues investigated in relation to quality assurance in the MTA context. Due to space limitation we are showing only some papers for each issue.

TABLE IV. Main issues about quality assurance of multi-tenant SaaS applications

	Description	Ref.
System Testing	Combinatorial testing: dynamic test sequences were used to achieve high architectural coverage	[8]
Regression Testing	Continuous testing with partitioning of data from tenants and generation of test case based on meta-data	[28]
Security	Framework for security management	[27]
	SecPlac: resource allocation model to support the security in the sharing of infrastructure among tenants	[29]
	TOSSMA (Tenant Oriented SaaS Security Management Architecture): an architecture to isolate resources for tenants through the injection of authorization controls	[30]
	Data combination privacy	[31]
QoS	MSSOptimiser (Multi-tenant SaaS Optimizer): an approach to select services addressing quality requirements	[32]

e) Database: Nineteen papers in SMS have cited database-related issues as a promising research direction. Saraswathi and Bhuvanewari [33] presented two alternatives for multi-tenant data architecture: i) one related to authentication and authorization and ii) a non-intrusive approach for large-scale applications. The authors described a process to apply them and guide engineers in the development of databases.

Maenhaut et al. [34] developed an approach for data management in a hierarchical way and taking into account some performance metrics. The main question addressed concerned the distribution of users and data into multiple instances of database. Yaish et al. [35] discussed an access control model based on a database schema. They also proposed an access control algorithm that enables users to access the data granted based on users groups or assigned roles.

2) RQ₂: What are the main research challenges and opportunities related to the development, testing and evolving of multi-tenant SaaS applications?

Although most studies have addressed the development of multi-tenant SaaS applications, standards are scattered, and do not often follow a methodical approach. Furthermore, the solutions are proprietary and rarely interoperable [25]. Traditional software testing cannot be applied to test applications in a Cloud environment due to it is designed for on-premise single-tenant applications [36].

Several issues should be considered during the testing of multi-tenant SaaS application: (i) resources are shared among tenants and their end-users, (ii) each variant application addresses a specific requirements set for a tenant, it is executed as if it was in a dedicated environment and can be composed of several components and (iii) a variant application is delivered to the customers through a run-time engine from cloud provider that weaves the tenant customization data and specific metadata to kernel code. Thus, each application provides different screens and logic.

According to Alkhatib et al. [25], the community still does not have effective quality metrics for the SaaS and new testing strategies are required to meet the challenges imposed by the cloud computing model. Software integration testing issues, validation methods and quality assurance standards addressing the interaction interfaces must be established. Since high system availability is essential to SaaS, the re-testing techniques considering the multi-tenancy feature are mandatory whenever software is changed for improvements or bug-fixing.

Recent studies on tests in cloud computing have addressed the verification of nonfunctional requirements as performance and security. Considering the selected studies, we have identified that research fields as (i) adjustments of conventional test criteria, (ii) test strategies for customization components, (iii) alternatives to verify the composition interfaces and impact new components, and (iv) approaches to regression test require more cooperation between industry and academia.

Regarding the evolution of multi-tenant SaaS applications, the community still has not provided well-defined approaches. The evolving activity of distributed systems may indicate guidelines for dealing with the isolated execution of instances of these applications.

3) *Research agenda*: In order to guide future research in the area of multi-tenancy, this section presents the major trends identified in this study, as follows:

Development process. Most of selected studies about development process proposed a solution or a process without a practical evaluation. Methodical approaches to guide the development of cloud-based applications require more research effort, especially taking into account the multi-tenancy.

Quality metrics. Metrics are used to guide managers during the software quality evaluation. Despite many studies mentioning their importance, there is still a lack of quality metrics for this context.

Testing activity. Few organizations and academy provide security testing, recovery testing, fault-tolerance testing or some alternative to cover the complexity of multi-tenancy for SaaS. In addition, there is a lack of standards to driven the development of interoperable test tools. From our point of view,

an ideal testing environment for multi-tenant SaaS systems needs to support the testing of a tenant specific application in runtime, without impacting others. For this, there are two main issues that should be carefully considered: (i) a variant application is generated through a dynamic compiler that combines the tenant specific metadata and customization data to kernel code and (ii) tenants can apply changes according to their concerns. Thus, it is important to ensure that the testing will not result in side effects to other tenants. A possible testing strategy is to generate the variant application that we want to test, perform its isolating and conduct the testing activity without making the other applications unavailable.

Continuous validation. Despite the on-demand software validation in the cloud environment includes the regression testing and frequent changes in the application, most of the regression test studies have focused on retest a version in a previously configured test environment. However, validation methods must be dynamic to deal with multi-tenancy.

Empirical studies to evaluate testing techniques and criteria. There is a lack for guidelines about which testing techniques and criteria to use considering the testing objectives in context of SaaS systems. It is necessary to know the usability, effectiveness and cost of these techniques and criteria. We have observed a lack of empirical studies to evaluate the use of techniques and testing criteria in cloud computing domain. In addition, many researchers argue that the traditional software testing cannot be satisfactorily applied to test cloud computing applications. This happens due to implicit characteristics of these applications, such as the high customization capabilities, dynamic environment and multi-tenancy.

Migrating process. Consolidated processes to guide the migration from web conventional to multi-tenant SaaS model can contribute with the adoption of this model for applications where only the multi-user model is not enough.

III. THREATS TO VALIDITY

Selection of primary studies. In order to ensure an unbiased selection process, research questions were defined and exclusion and inclusion criteria were specified for the obtaining of relevant studies. However, threats cannot be ruled from a quality evaluation perspective, although the studies were selected by a score assignment based on relevance.

Relevant primary studies not selected. Although many sources were used for the selection of primary studies, some might have been neglected. We tried to reduce this threat by selecting sources that index studies from the main scientific sources in cloud computing and covering most of the relevant papers.

Reviewers reliability. All reviewers that participated in this study work on cloud computing. The protocol was assessed by specialists, so that deviations during the analysis could be avoided.

Data extraction. It is worth mentioning that not all information was obvious to answer the research questions. Several sources, as external papers and technical reports were consulted, so that the validity of the process could be ensured.

In case of disagreement among the reviewers, a specialist was called to guarantee the correct decision.

IV. CONCLUDING REMARKS

Software as a service is a way of delivering applications over the Internet as a service for multiple customers. From the point of view of service providers, the computing resources to be offered must be broadly shared. For the users, it is important to customize the application according to their specific requirements. In this scenario, an architectural pattern called multi-tenancy is gaining more ground in the application space on cloud.

In order to provide a mapping of research topics on the multi-tenancy of SaaS systems and identifying new research opportunities, we have conducted an SMS in which 89 primary studies were selected for discussions. We have defined two research questions that reflect the scope of the study and five categories to map the contributions and challenges.

This mapping study also points out the need for experimental studies evaluating the proposed approaches and a systematic test strategy extending different techniques to increase the quality of these applications. In our future research, we intend to compare the evidence identified in this work with evidence from industrial cloud projects in order to define new hypotheses, which will guide the definition of approaches for testing of multi-tenant SaaS applications.

REFERENCES

- [1] J. Ru and J. Keung, "An empirical investigation on the simulation of priority and shortest-job-first scheduling for cloud-based software systems," in *Software Engineering Conference*. IEEE, 2013, pp. 78–87.
- [2] S. Tai, J. Nimis, A. Lenk, and M. Klems, "Cloud service engineering," in *32nd Int. Conf. on Software Engineering*, 2010, pp. 475–476.
- [3] P. Mell and T. Grance, "The nist definition of cloud computing," NIS, Tech. Rep., 2010.
- [4] L. M. Vaquero, L. Roderio-Merino, J. Caceres, and M. Lindner, "A break in the clouds," *ACM Computer Communication Review*, vol. 39, no. 1, p. 50, 2008. [Online]. Available: <http://dl.acm.org/citation.cfm?id=1496091.1496100>
- [5] I. Chana and P. Chawla, "Testing perspectives for cloud-based applications," in *Software Engineering Frameworks for the Cloud Computing Paradigm*. Springer, 2013, pp. 145–164.
- [6] C.-P. Bezemer and A. Zaidman, "Multi-tenant saas applications: Maintenance dream or nightmare?" in *Work. on Software Evolution and Int. Work. on Principles of Software Evolution*, 2010, pp. 88–92.
- [7] B. Sengupta and A. Roychoudhury, "Engineering multi-tenant software-as-a-service systems," in *3rd Int. Work. on Principles of Engineering Service-Oriented Systems*, 2011, pp. 15–21.
- [8] W.-T. Tsai, Q. Li, C. J. Colbourn, and X. Bai, "Adaptive fault detection for testing tenant applications in multi-tenancy saas systems," in *IEEE Int. Conf. on Cloud Engineering*, 2013, pp. 183–192.
- [9] B. Kitchenham, "Procedures for performing systematic reviews," Keele University, Tech. Rep., 2004.
- [10] S. Kang, J. Myung, J. Yeon, S.-w. Ha, T. Cho, J.-m. Chung, and S.-g. Lee, "A general maturity model and reference architecture for saas service," in *Database Systems for Advanced Apps.*, 2010, pp. 337–346.
- [11] J. Ru, J. Grundy, and J. Keung, "Software engineering for multi-tenancy computing challenges and implications," in *Int. Work. on Innovative Soft. Dev. Methodologies and Practices*, 2014, pp. 1–10.
- [12] E. Truyen, N. Cardozo, S. Walraven, J. Vallejos, E. Bainomugisha, S. Günther, T. D'Hondt, and W. Joosen, "Context-oriented programming for customizable SaaS applications," in *27th ACM symposium on applied computing*, 2012, pp. 418–425.
- [13] S. Walraven, D. Van Landuyt, E. Truyen, K. Handekyn, and W. Joosen, "Efficient customization of multi-tenant software-as-a-service applications with service lines," *JSS*, vol. 91, pp. 48–62, 2014.
- [14] S. S. Kale and R. H. Borhade, "Development of multitenant saas framework at single instance and with zero effort multitenancy," in *Advances in Computing, Communications and Informatics (ICACCI)*. IEEE, 2013, pp. 834–839.
- [15] T. Nam and K. Yeom, "Ontology model to support multi-tenancy in software as a service environment," in *Int. Conf. on Future Internet of Things and Cloud*, 2014, pp. 146–151.
- [16] A. M. Manduca, E. V. Munson, R. P. Fortes, and M. G. C. Pimentel, "A nonintrusive approach for implementing single database, multitenant services from web applications," in *29th ACM Symposium on Applied Computing*, 2014, pp. 751–756.
- [17] P. Morakos and A. Meliones, "Design and implementation of a cloud saas framework for multi-tenant applications," in *5th Int. Conf. on Information, Intelligence, Systems and Applications*, 2014, pp. 273–278.
- [18] H. Sun, T. Zhao, Y. Tang, and X. Liu, "A qos-aware load balancing policy in multi-tenancy environment," in *8th Int. Symposium on Service Oriented System Engineering*, 2014, pp. 140–147.
- [19] H. J. Moon, H. Hacigumus, Y. Chi, and W.-P. Hsiung, "Swat: A lightweight load balancing method for multitenant databases," in *16th Int. Conf. on Extending Database Technology*, 2013, pp. 65–76.
- [20] T. Patikirikorala, I. Kumara, A. Colman, J. Han, L. Wang, D. Weerasiri, and W. Ranasinghe, "Dynamic performance management in multi-tenanted business process servers using nonlinear control," in *Int. Conf. on Service-Oriented Computing*, 2012, pp. 206–221.
- [21] R. Krebs, A. Wert, and S. Kounev, "Multi-tenancy performance benchmark for web application platforms," in *Int. Conf. on Web Eng.*, 2013, pp. 424–438.
- [22] W.-T. Tsai and X. Sun, "SaaS multi-tenant application customization," in *IEEE 7th Int. Symposium on Service Oriented System Engineering*, 2013, pp. 1–12.
- [23] L. Ramachandran, N. C. Narendra, and K. Ponnalagu, "Dynamic provisioning in multi-tenant service clouds," *Service Oriented Computing and Applications*, vol. 6, no. 4, pp. 283–302, 2012.
- [24] S. Walraven, E. Truyen, and W. Joosen, "A middleware layer for flexible and cost-efficient multi-tenant applications," in *Int. Conf. on Middleware*. Springer, 2011, pp. 370–389.
- [25] H. Alkhatib, P. Faraboschi, E. Frachtenberg, H. Kasahara, D. Lange, P. Laplante, A. Merchant, D. Milojevic, and K. Schwan, "IEEE CS 2022 report," IEEE Computer Society, Tech. Rep., 2014.
- [26] K. Wood and M. Anderson, "Understanding the complexity surrounding multitenancy in cloud computing," in *IEEE 8th Int. Conf. on e-Business Engineering*, 2011, pp. 119–124.
- [27] M. Almorsy, J. Grundy, and A. S. Ibrahim, "Collaboration-based cloud computing security management framework," in *IEEE Int. Conf. on Cloud Computing*, 2011, pp. 364–371.
- [28] W.-T. Tsai, Q. Shao, Y. Huang, and X. Bai, "Towards a scalable and robust multi-tenancy SaaS," in *Asia-Pacific Symp. on Internetware*, 2010, p. 8.
- [29] E. Saleh, J. Sianipar, I. Takouna, and C. Meinel, "Secplace: A security-aware placement model for multi-tenant SaaS environments," in *Int. Conf. on Ubiquitous Intell. and Comp.*, 2014, pp. 596–602.
- [30] M. Almorsy, J. Grundy, and A. S. Ibrahim, "Tossm: A tenant-oriented saas security management architecture," in *IEEE 5th Int. Conf. on Cloud Computing*, 2012, pp. 981–988.
- [31] K. Zhang, Q. Li, and Y. Shi, "Data privacy preservation during schema evolution for multi-tenancy applications in cloud computing," in *Web Information Systems and Mining*. Springer, 2011, pp. 376–383.
- [32] Q. He, J. Han, Y. Yang, J. Grundy, and H. Jin, "Qos-driven service selection for multi-tenant saas," in *Int. Conf. on Cloud Computing*, 2012, pp. 566–573.
- [33] M. Saraswathi and T. Bhuvaneshwari, "Multitenant SaaS model of cloud computing: Issues and solutions," in *Communication and Network Technologies*. IEEE, 2014, pp. 27–32.
- [34] P.-J. Maenhaut, H. Moens, M. Decat, J. Bogaerts, B. Lagaisse, W. Joosen, V. Ongenaes, and F. De Turck, "Characterizing the performance of tenant data management in multi-tenant cloud authorization systems," in *Network Operations and Management Symposium (NOMS), IEEE*, 2014, pp. 1–8.
- [35] H. Yaish and M. Goyal, "A multi-tenant database architecture design for software applications," in *ICCSE*, 2013, pp. 933–940.
- [36] Z. Mahmood and S. Saeed, *Software engineering frameworks for the cloud computing paradigm*. Springer, 2013.

Applying Verbal Decision Analysis to Task Allocation in Distributed Development of Software

Marum Simão Filho

Graduate Program in Applied Computer Science
University of Fortaleza / 7 de Setembro College
Fortaleza, Brazil
marum@unifor.br

Plácido R. Pinheiro, Adriano B. Albuquerque

Graduate Program in Applied Computer Science
University of Fortaleza
Fortaleza, Brazil
{placido, adrianoba}@unifor.br

Abstract—The management of distributed software development projects presents many challenges. One of them happens right at the start of the project and consists of the allocation of tasks between remote teams. When allocating a task to a site, the project manager takes into account several factors such as technical knowledge of staff and proximity to the client. The project manager usually takes this decision in a subjective way. The verbal decision analysis is an approach based on solving problems through multi-criteria qualitative analysis, which means it considers the analysis of subjective criteria. This paper describes the application of verbal decision analysis methodologies ORCLASS and ZAPROS III-i to classify and rank the most relevant factors that the project managers should take into account when allocating tasks in projects of distributed development of software.

Keywords — *Distributed Development of Software, Task Allocation, Verbal Decision Analysis, ORCLASS, ZAPROS III-i.*

I. INTRODUCTION

The Distributed Development of Software (DDS) is a reality more and more present in modern companies. The perspective to expand the workforce capacity, the conquest of new markets and the cost reduction possibility are some of the reasons that make software development companies adopt distributed development [27]. On the other hand, the distribution brings many challenges, such as language and time zone differences and increased complexity of coordinating and controlling the project [25]. In this context, the allocation of tasks is an even more critical activity for project planning because of the distribution [24]. The distribution of tasks to remote teams can be seen as a fundamental activity for the success of a distributed project. However, this activity is still a major challenge in global software development due to limited understanding of the factors that influence task allocation decisions [8].

Deciding which task we should allocate for each team is typically a decision-making problem. Routinely, the project manager makes this decision based on their experience and knowledge about the project and the teams involved. We mean that a high degree of subjectivity is present in the decision-making process. This is an appropriate setting for Verbal decision analysis (VDA), which consist of an approach based on multicriteria problem solving through its qualitative analysis [17], i.e., VDA methods take into consideration the criteria's subjectivity.

This paper describes the application of a hybrid methodology using VDA methods to classify and rank order the most relevant factors to be considered by project managers when allocating tasks in projects of distributed development of software. Firstly, expert interviews were conducted to identify the criteria and the criteria values. Then, a questionnaire was applied to a group of project managers to characterize each factor through the criteria and criteria values. Next, the ORCLASS method was applied to divide the factors into preference groups. Finally, the ZAPROS III-i was then applied to rank order the preferable factors.

The rest of the paper is organized as follows: Section II shortly deals with issues involving task allocation in distributed development of software. Section III provides a brief description of the verbal decision analysis methods ORCLASS and ZAPROS III-i. Section IV describes the application of the hybrid methodology using ORCLASS and ZAPROS III-i. Section V presents the results of our work. Finally, in Section VI, we provide the conclusions and suggestions for further work.

II. TASK ALLOCATION IN DDS

The allocation of tasks is a critical activity for any kind of project, especially in a distributed scenario. Most of the time, few factors drive the allocation of tasks, such as hand labor costs. Risks and other relevant factors such as the workforce skills, innovation potential of different regions, or cultural factors are often insufficiently recognized [12].

Many studies about the tasks allocation in DDS have been carried out along the years aiming at mapping this topic and its features. Lamersdorf et al. [12] developed an analysis of the existing approaches to distribution of duties. The analysis was comprehensive and involved procedures for the distributed development, distributed generation, and distributed systems areas. Lamersdorf et al. [13] conducted a survey on the state of practice in DDS in which they investigated the criteria that influence task allocation decisions. Lamersdorf and Münch [11] presented TAMRI (Task Allocation based on Multiple cRiteria), a model based on multiple criteria and influencing factors to support the systematic decision of task allocation in distributed development projects.

Ruano-Mayoral et al. [31] presented a methodological framework to allocate work packages among participants in global software development projects. Marques et al. [25] performed a systematic mapping, which enabled us to identify models that propose to solve the problems of allocation of

tasks in DDS projects. They intended to propose a combinatorial optimization-based model involving classical task scheduling problems. Marques et al. [24] also performed a tertiary review applying the systematic review method on systematic reviews that address the DDS issues.

Galviņa and Šmite [9] provided an extensive literature review for understanding the industrial practice of software development processes and concluded that the evidence of how these projects are organized is scarce. Babar and Zahedi [3] presented a literature review considering the studies published in the International Conference in Global Software Engineering (ICGSE) between 2007 and 2011. It was found that the vast majority of the evaluated studies were in software development governance and its sub-categories, and much of the work had focused on the human aspects of the GSD rather than technical aspects.

Almeida et al. [1] presented a multi-criteria decision model for planning and fine-tuning such project plans: Multi-criteria Decision Analysis (MCDA). The model was developed using cognitive mapping and MACBETH (Measuring Attractiveness by a Categorical Based Evaluation Technique) [4]. In [2], Almeida et al. applied (MCDA) on the choice of DDS Scrum project plans that have a better chance of success.

Simão Filho et al. [32] conducted a quasi-systematic review of studies of task allocation in DDS projects that incorporate agile practices. The study brought together a number of other works, allowing the establishment of the many factors that influence the allocation of tasks in DDS. These factors are very important for this work and will be mentioned later.

III. VERBAL DECISION ANALYSIS

Decision-making is an activity that is part of people's and organizations' lives. In most problems, to make a decision, a situation is assessed against a set of characteristics or attributes, i.e., it involves the analysis of several factors, also called criteria. When a decision can generate a considerable impact, such as management decisions, and must take into account some factors, the use of methodologies to support the decision-making process is suggested, because choosing the inappropriate alternative can lead to waste of resources, time, and money, affecting the company.

The decision-making scenario that involves the analysis of alternatives from several viewpoints is called multi-criteria decision analysis and is supported by multicriteria methodologies [4]. These methodologies favor the generation of knowledge about the decision context, which helps raise the confidence of the decision maker [8 and 23].

The verbal decision analysis is an approach to solving multicriteria problems through qualitative analysis [15]. The VDA supports the decision-making process through the verbal representation of problems. Some examples of the application of VDA in real problems are given next. In [20], Machado et al. applied VDA to CMMI practices. In [26 and 40], the authors used VDA in digital TV applications. In [21], Machado applied a hybrid model of VDA in selecting project management approaches. In [37], Tamanini et al. proposed a VDA-based model to cashew chestnut industrialization process. In [5, 6, 7, 38 and 39], the authors developed studies

applying VDA to the diagnosis of Alzheimer's disease. In [41], Tamanini and Pinheiro approached the incomparability problem on ZAPROS. In [30], Pinheiro et al. studied multicriteria on the learning process in software engineering.

According to [28], the traditional methods of VDA aimed at solving problems with a lot of alternatives and a limited number of criteria and criteria values, since they were designed for the construction of a general rule for the decision, regardless of which alternatives belonged to the real alternatives set. However, this characteristic has changed recently, and new methods that elicit the preferences based on the real alternatives to the problem have been proposed.

The VDA methodologies can be used for ordering or sorting the alternatives. Among the classification methods, we can mention ORCLASS, SAC, DIFCLASS, and CYCLE. Some sorting methods are PACOM, ARACE, and those from ZAPROS family (ZAPROS-LM, STEPZAPROS, ZAPROS III and III-i) [34]. Fig. 1 shows the VDA classification and ordering methods.

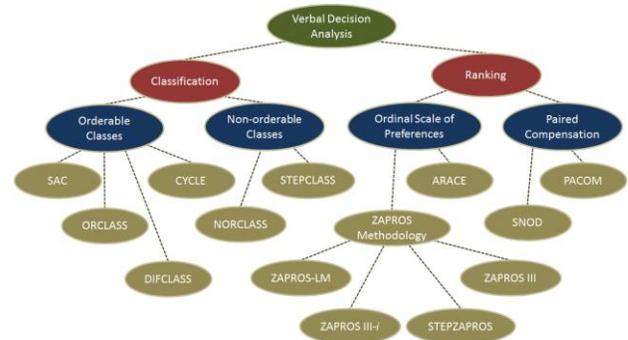


Fig. 1. VDA Methods for Classification and Ordering [34]

A. The ORCLASS Method for Classification

ORCLASS methodology aims at classifying the alternatives in a given set: the decision maker needs these alternatives to be categorized into a small number of decision classes or groups, usually two. The first group covers the most preferable alternatives, and the less preferable alternatives belong to the second one [17].

The flowchart with steps to apply the ORCLASS method was presented in [35]. In that scheme, the application of the ORCLASS method can be divided into three stages: Problem Formulation, Structuring of the Classification Rule and Analysis of the Information Obtained. In the first stage, the problem's formulation, the set of criteria, criteria values, and the decision groups are defined. The criteria values must be sorted in an ascending order of preference (from most to least preferable). In case of three criteria and three criteria values, we have the following criteria values for each criterion: A1, A2 and A3 (for criterion A), B1, B2 and B3 (for criterion B), and C1, C2 and C3 (for criterion C). A1, B1 and C1 are the most preferable values, and A3, B3 and C3 are the least preferable values for criteria A, B and C, respectively. In this case, possible alternatives are represented as [A1, B2, C3] and [A3, B1, C2].

Then, the construction of the classification rule is carried out based on the decision maker's preferences. We use the

same concepts presented in [17], based on which a classification task is presented as a set of boards. Each cell of the board is composed of a combination of values for each criterion defined for the problem, which represents a possible alternative to the problem [23]. It should be noted that the most preferable hypothetical alternative naturally belongs to the first group (A1B1C1) as long as the least preferable one belongs to the second group (A3B3C3).

To facilitate the decision-making process, Machado et al. developed a tool called ORCLASSWEB, and made it available for use over the Internet (<http://www2.unifor.br/OrclassWeb>) [22]. The ORCLASSWEB tool proposes to automate the comparison process of alternatives and to provide the decision maker a concrete result for the problem, according to ORCLASS definition. Other applications of the ORCLASS method can be found in [10, 19 and 23].

B. The ZAPROS III-i Method for Rank Ordering

The ZAPROS methodology aims at ranking multi-criteria alternatives in scenarios involving a rather small set of criteria and criteria values, and a great number of alternatives. It is structured in three stages: Problem Formulation, Elicitation of Preferences and Comparison of Alternatives.

In ZAPROS LM method, we carry out the elicitation of preferences by comparing vectors of alternatives [17]. The ZAPROS III method [14] is an evolution of the ZAPROS LM. There are modifications that make it more efficient and more accurate on inconsistencies. The subjectivity and the qualitative aspect of ZAPROS method can cause losses to the method's comparison capacity and make the incomparability cases between the alternatives unavoidable [16]. ZAPROS III-i introduces modifications in the comparison process of alternatives so that it could minimize or even eliminate the incomparability problem of the ZAPROS method [43].

The ZAPROS III-i method's flowchart to rank order a set of alternatives can be found in [35]. In the first stage, we obtain the relevant criteria and their values to the decision-making process. In the second stage, we generate the scale of preferences based on the decision maker's preference. In the last stage, we perform the comparison between the alternatives based on the decision maker's preferences. Trying to reduce the number of incomparability cases, we apply the same structure proposed in [14], but the comparison of pairs of the alternatives' substage was modified according to the one proposed in [29].

IV. THE HYBRID METHODOLOGY APPLICATION

To classify and rank order the most important factors that project managers should consider when allocating tasks in projects of distributed development of software, we applied a hybrid methodology, which consists of five main steps, as follows:

- A - Identification of the Influencing Factors;
- B - Definition of the Criteria and Criteria values;
- C - Definition of the Alternatives, Decision Groups, and Alternatives' Characterization;
- D - The ORCLASS Method Application; and
- E - The ZAPROS-III-i Method Application.

In the next subsections, we explain each of the steps.

A. Identification of the Influencing Factors

First, we conducted a literature research to identify the main influencing factors that should be considered when allocating tasks in projects of distributed development of software. Table 1 shows the factors found as a result of this research, and that worked as the alternatives to our decision problem [35].

TABLE 1. INFLUENCING FACTORS ON TASK ALLOCATION IN DDS PROJECTS

ID	Alternatives
Factor1	Technical expertise
Factor2	Expertise in business
Factor3	Project manager maturity
Factor4	Proximity to client
Factor5	Low turnover rate
Factor6	Availability
Factor7	Site maturity
Factor8	Personal trust
Factor9	Time zone
Factor10	Cultural similarities
Factor11	Willingness at site

B. Definition of the Criteria and Criteria values

Next, we interviewed a group of 4 project management experts to define the criteria and the criteria values. This is the definition stage of the criteria. For each criterion, we established a scale of values associated with it [18, 19 and 40]. The criteria values were ordered from the most preferable value to the least preferable one. As result of this step, we got the list of criteria and criteria values for the problem of selecting the most important factors to be considered in task allocation in DDS projects, which is listed next [33]:

1. Criterion A: Facility for carrying out the task remotely.
 - A1. It facilitates much: The implementation of the remote task is much easier if the factor is present.
 - A2. It facilitates: The implementation of the remote task is easier if the factor is present.
 - A3. Indifferent: The presence of the factor is indifferent to the implementation of the remote task.
2. Criterion B: Time for the project.
 - B1. High gain: The presence of the factor can cause much reduction of the period referred to perform the task.
 - B2. Moderate gain: The presence of the factor may cause some reduction of the time limit for performing the task.
 - B3. No gain: The presence of the factor does not cause changes to the deadline to execute the task.
3. Criterion C: Cost for the project.
 - C1. High gain: The presence of the factor can cause a lot of cost reduction expected to perform the task.
 - C2. Moderate gain: The presence of the factor may cause some reduction of the time limit for performing the task.
 - C3. No gain: The presence of factor induces no change compared to the estimated cost to perform the task.

C. Definition of the Alternatives, Decision Groups, and Alternatives' Characterization

We created a questionnaire to gather information and opinions about the factors that influence the allocation of tasks

in DDS projects. We applied the questionnaire to the Web to a group of 20 project managers and consisted of two parts. The first part aimed to trace the respondents profile about his/her professional experience and education.

The second part of the questionnaire inquired the views of experts on the factors that influence the allocation of tasks in DDS projects. For our problem, we described such influencing factors as alternatives. Thus, in every question, the professional analyzed the influencing factors about a set of criteria and criteria values and selected what criterion value that best fitted the factor analyzed. An example of question is as follows:

1. Factor: Technical expertise – knowledge of the techniques, languages, frameworks, tools, APIs, etc. needed by the team to accomplish the task.

(a) Criterion A: Facility for carrying out the task remotely
 A1. It facilitates much. A2. It facilitates. A3. Indifferent.

(b) Criterion B: Time for the project
 B1. High gain. B2. Moderate gain. B3.No gain.

(c) Criterion C: Cost for the project
 B1. High gain. B2. Moderate gain. B3.No gain.

We did the same for the other ten factors. Then, we analyzed the responses to determine the criteria values representing the alternatives. For each influencing factor, we filled the final table based on the replies of the majority of professionals. We then selected the value of the criterion that had the greatest number of choices to represent the alternative. Table 2 summarizes the responses to the questionnaire, showing the sum of the answers and characterization of alternative according to the values of each criterion (represented in the “Final Vector” column). The bold numbers in gray cells in the table indicate the criteria values selected by most of the interviewed professionals to represent a certain factor.

TABLE 2. CHARACTERIZATION OF ALTERNATIVES [33]

Criteria/ alternatives	Facility for carrying out the task remotely			Time for the project			Cost for the project			Final vector
	A1	A2	A3	B1	B2	B3	C1	C2	C3	
Factor1	11	7	2	13	6	1	11	7	2	A1B1C1
Factor2	15	3	2	13	7	0	10	8	2	A1B1C1
Factor3	8	11	1	5	14	1	7	10	3	A2B2C2
Factor4	13	4	3	8	10	2	8	10	2	A1B2C2
Factor5	14	6	0	15	4	1	12	7	1	A1B1C1
Factor6	10	8	2	13	5	2	9	6	5	A1B1C1
Factor7	16	3	1	11	9	0	9	11	0	A1B1C2
Factor8	8	10	2	6	11	3	3	13	4	A2B2C2
Factor9	3	12	5	3	8	9	3	6	11	A2B3C3
Factor10	4	13	3	3	10	7	3	8	9	A2B2C3
Factor11	10	8	2	10	8	2	9	6	5	A1B1C1

We emphasize that the various answers given by professionals, considering they have experienced project managers, were related to the fact that they have different professional backgrounds. Thereby, the characterization of a particular factor was based on answers given by most professionals.

Thus, the decision groups were defined as follows. Group I: The influencing factors that will be selected as the most important ones that project managers should take into account when allocating tasks to remote teams (preferable factors).

Group II: The influencing factors that should be less considered by project managers when they need to allocate tasks to remote teams (not preferable factors).

D. The ORCLASS Method Application

The ORCLASS method application was aided by the ORCLASSWEB tool, which was divided into four steps: Criteria and criteria value definition; Alternatives definition; Construction of the classification rule; and Results Generation.

We introduced the problem’s criteria into the ORCLASSWEB tool. In this step, we specified the criteria’s names and their possible values. The tool allowed us to insert all the necessary criteria. Next, we introduced the problem’s alternatives into the ORCLASSWEB tool. The tool allowed us to inform the alternatives’ names, and their representations in criteria values, according to the criteria defined in the previous step (and in the column “final vector” in Table 2).

The ORCLASSWEB tool also supported the construction of the classification rule. The tool calculates which question would be the next one that the decision maker is supposed to answer according to the ORCLASS method’s rules for the selection of the most significant alternative. In this step, we had the support of an experienced project manager to answer the questions to classify the alternatives. The classification rule was completed based on the decision-maker choices. In the end, the tool processed the full classification of the alternatives.

As result of applying the ORCLASSWEB tool, we got the following factors to compose the Group I (the preferable factors): Factor1 - Technical expertise, Factor2 - Expertise in business, Factor3 - Project manager maturity, Factor4 - Proximity to client, Factor5 - Low turnover rate, Factor6 - Availability, Factor7 - Site maturity, Factor11 - Willingness at site, and Factor8 - Personal trust. They are the most important ones that project managers should consider when allocating tasks in projects of distributed development of software, according to the ORCLASS method. In the Group II, which was composed of the least preferable factors, we got the following factors: Factor9 - Time zone and Factor10 - Cultural similarities.

E. The ZAPROS-III-i Method Application

After determining the preferred factors using the OSCLASS method, we moved on to the stage of ordering. At this stage, we applied the ZAPROS III-i method to put in order the preferable factors, such that it is possible to establish a ranking of preferred factors. In this step, the least preferable factors were discarded, thereby reducing our workspace.

To facilitate the decision-making process and perform it consistently, we used the ARANAÚ tool, presented in [35, 42 and 43]. The tool, which was implemented in Java platform, was first developed in [36] to support ZAPROS III method. In this work, we used the updated version to ZAPROS III-i method. The use of ZAPROS III-i method in the ARANAÚ tool requires four steps, as follows: Criteria and criteria value definition; Preferences Elicitation; Alternatives Definition; and Results Generation.

First of all, we introduced the criteria presented in the problem into the ARANAÚ tool. Next, the decision-maker

decides the preferences. The interface for elicitation of preferences presents questionings that can be easily answered by the decision-maker to obtain the scale of preferences. The process occurs in two stages: elicitation of preferences for quality variation of the same criteria and elicitation of preferences between pairs of criteria. The questions provided require a comparison considering the two reference situations [35]. Once the scale of preferences is structured, the next step is to define the problem's alternatives. The alternatives to our problem are the preferable factors integrating of Group I.

V. RESULTS

After introducing all the data and answering the necessary questions, the decision maker is presented with the result in a table containing the alternatives and their criteria evaluations, formal index of quality and rank, as exposed in Table 3. Note that there are five alternatives (factors) that are in the same ranking position (first position), and their FIQ's values are equals to zero. This occurs because all of them got the best evaluation according to the survey filled out by the professionals (A1, B1, C1), which is the best possible evaluation.

TABLE 3. THE FINAL RANKING OF ALTERNATIVES

Rank	Alternative	Representation	FIQ
1	Factor1 - Technical expertise	A1B1C1	0
1	Factor2 - Expertise in business	A1B1C1	0
1	Factor5 - Low turnover rate	A1B1C1	0
1	Factor6 - Availability	A1B1C1	0
1	Factor11 - Willingness at site	A1B1C1	0
2	Factor7 - Site maturity	A1B1C2	6
3	Factor4 - Proximity to client	A1B2C2	10
4	Factor3 - Project manager maturity	A2B2C2	11
5	Factor8 - Personal Trust	A2B2C2	11

A graph showing the dominance relations between the alternatives is also generated by the ARANAÚ tool and is exposed to provide a more detailed analysis of the problem's resolution. This graph can be seen in Fig. 2.

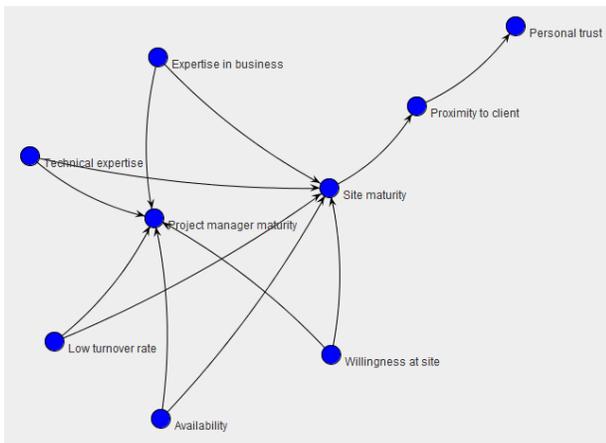


Fig. 2. The graph showing the dominance relations between the alternatives

VI. CONCLUSION AND FUTURE WORKS

For large software development projects, working with distributed teams has been an alternative increasingly present in large companies. However, the distribution brings many challenges, particularly concerning the allocation of tasks among remote teams, since there are many factors that project

managers should take into consideration. Typically, this multi-criteria decision-making problem involves subjective aspects. The verbal decision analysis methods support decision-making process through multi-criteria qualitative analysis.

The main contribution of this work was to apply a hybrid methodology based on ORCLASS and ZAPROS III-I methods to select and rank order the most important factors that project managers should consider when allocating tasks among distributed teams. Two tools, ORCLASSWEB, and ARANAÚ, supported this work allowing the performance of the tasks in a fast and practical way. Previously, we conducted interviews and applied questionnaires to a group of project management experts so that we could identify the factors, the criteria and the criteria values to use in the methods.

As future work, we intend to combine this hybrid methodology to other decision support methods so that we can compare the results of them. Also, we intend to develop case studies with real life situations to verify the methodology. Finally, we propose to apply VDA methods to help choosing the team that should be assigned a specific task, based on the task characteristics and teams profiles.

ACKNOWLEDGMENT

The first author is thankful for the support given by the “Coordination for the Improvement of Higher Level-or Education- Personnel” (CAPES) and 7 de Setembro College. The second author is grateful to National Council of Technological and Scientific Development (CNPq) via Grants #305844/2011-3. The authors would like to thank The Edson Queiroz Foundation/University of Fortaleza for all the support.

REFERENCES

- [1] Almeida L.H., Albuquerque, A.B, Pinheiro, P.R., “A Multi-criteria Model for Planning and Fine-Tuning Distributed Scrum Projects”, In: Proceedings of the 6th IEEE International Conference on Global Software Engineering, 2011.
- [2] Almeida L.H., Albuquerque, A.B, Pinheiro, P.R., “Applying Multi-Criteria Decision Analysis to Global Software Development with Scrum Project Planning”, Lecture Notes in Computer Science, v. 6954, p. 311-320, 2011.
- [3] Babar, M. A. and Zahedi, M., “Global Software Development: A Review of the State-Of-The-Art (2007 – 2011)”, IT University Technical Report Series. IT University of Copenha-gen, 2012.
- [4] Bana e Costa, C.A., Sanchez-Lopez, R., Vansnick, J.C., De Corte, J.M., “Introducción a MACBETH”, In: Leyva López, J.C. (ed.). Análisis Multicriterio para la Toma de Decisiones: Métodos y Aplicaciones, Plaza y Valdés, México (233-241), 2011.
- [5] Castro, A.K.A., Pinheiro, P.R., Pinheiro, M.C.D., “Applying a Decision Making Model in the Early Diagnosis of Alzheimer’s Disease”, Rough Sets and Knowledge Technology, Lecture Notes in Computer Science, v.4481, p.149–156, 2007.
- [6] Castro, A.K.A., Pinheiro, P.R., Pinheiro, M.C.D., “A Multicriteria Model Applied in the Diagnosis of Alzheimer’s Disease”, Rough Sets and Knowledge Technology, Lecture Notes in Computer Science, v.5009, p.612–619, 2008.
- [7] Castro, A.K.A., Pinheiro, P.R., Pinheiro, M.C.D., Tamanini, I., “Applied Hybrid Model in the Neuropsychological Diagnosis of the Alzheimer’s Disease: A Decision Making Study Case”, International Journal of Social and Humanistic Computing (IJSHC), v.1, n.3, p.331–345, 2010. DOI:10.1504/IJSHC.2010.032692.
- [8] Evangelou, C., Karacapilidis, N. and Khaled, O.A., “Interweaving knowledge management, argumentation and decision making in a collaborative setting: the KAD ontology model”, International Journal of Knowledge and Learning 1, 1/2, 130–145, 2005.

- [9] Galviņa, Z. and Šmite, D., “Software Development Processes in Globally Distributed Environment”, In: Scientific Papers, University Of Latvia, Vol. 770, Computer Science and Information Technologies, 2011.
- [10] Gomes, L.F.A.M., Moshkovich, H., Torres, A., “Marketing decisions in small businesses: how verbal decision analysis can help”, *International Journal Management and Decision Making* 11, 1, 19–36, 2010.
- [11] Lamersdorf, A. and Münch, J., “A multi-criteria distribution model for global software development projects”, *The Brazilian Computer Society*, 2010.
- [12] Lamersdorf, A., Münch, J. and Rombach, D., “Towards a Multi-Criteria Development Distribution Model: An Analysis of Existing Task Distribution Approaches”, In: *IEEE International Conference on Global Software Engineering, ICGSE 2008*, 2008.
- [13] Lamersdorf, A., Münch, J. and Rombach, D., “A Survey on the State of the Practice in Distributed Software Development: Criteria for Task Allocation”, In *Fourth IEEE International Conference on Global Software Engineering, ICGSE 2009*, 2009.
- [14] Larichev, O.I., “Ranking multicriteria alternatives: The Method ZAPROS III”, *European Journal of Operational Research*, v. 131, n. 3, p. 550–558, 2001.
- [15] Larichev, O.I., Brown, R., “Numerical and verbal decision analysis: comparison on practical cases”, *Journal of Multicriteria Decision Analysis* 9, 6, 263–273, 2000.
- [16] Larichev, O., “Method ZAPROS for Multicriteria Alternatives Ranking and the Problem of Incomparability”, *Informatica*, v. 12, n. 1, p. 89–100, 2001.
- [17] Larichev, O.I. and Moshkovich, H.M., “Verbal Decision Analysis for Unstructured Problems”, Boston: Kluwer Academic Publishers, 1997.
- [18] Machado, T.C.S., Menezes, A.C., Pinheiro, L.F.R., Tamanini, I., Pinheiro, P.R., “The selection of prototypes for educational tools: an applicability in verbal decision analysis”, *IEEE International Joint Conferences on Computer, Information, and Systems Sciences, and Engineering*, 2010.
- [19] Machado, T.C.S., Menezes, A.C., Pinheiro, L.F.R., Tamanini, I., Pinheiro, P.R., “Applying verbal decision analysis in selecting prototypes for educational tools”, *IEEE International Conference on Intelligent Computing and Intelligent Systems*, Xiamen, China, pp. 531–535, 2010.
- [20] Machado, T.C.S., Pinheiro, P.R., Albuquerque, A.B., de Lima, M.M.L., “Applying verbal decision analysis in selecting specific practices of CMMI”, *Lecture Notes in Computer Science* 7414, 215–221, 2012.
- [21] Machado, T. C. S., “Towards Aided by Multicriteria Support Methods and Software Development: A Hybrid Model of Verbal Decision Analysis for Selecting Approaches of Project Management”, Master Thesis. Master Program in Applied Computer Sciences, University of Fortaleza, 2012.
- [22] Machado, T. C. S., Pinheiro, P. R., Tamanini, I., “OrclassWeb: A Tool Based on the Classification Methodology ORCLASS from Verbal Decision Analysis Framework”, *Mathematical Problems in Engineering*, v. 2014, Article ID 238168, 11 pages, DOI: 10.1155/2014/238168, 2014.
- [23] Machado, T.C.S., Pinheiro, P.R., Tamanini, I., “Project Management Aided by Verbal Decision Analysis Approaches: A Case Study for the Selection of the Best SCRUM Practices”, *International Transactions in Operational Research*, v. 22, n. 2, p.287-312, DOI: 10.1111/itor.12078, 2014.
- [24] Marques, A. B., Rodrigues, R., and Conte, T., “Systematic Literature Reviews in Distributed Software Development: A Tertiary Study”, In: *IEEE International Conference on Global Software Engineering, ICGSE 2012*: 134-143, 2012.
- [25] Marques, A. B., Rodrigues, R., Prikladnicki R. and Conte, T., “Alocação de Tarefas em Projetos de Desenvolvimento Distribuído de Software: Análise das Soluções Existentes”, *II Congresso Brasileiro de Software, V WDDS – Workshop de Desenvolvimento Distribuído de Software*, São Paulo, 2011.
- [26] Mendes, M.S., Carvalho, A.L., Furtado, E., Pinheiro, P.R., “A co-evolutionary interaction design of digital TV applications based on verbal decision analysis of user experiences”, *International Journal of Digital Culture and Electronic Tourism* 1, 312–324, 2009.
- [27] Miller, A., “Distributed Agile Development at Microsoft patterns & practices”, *Microsoft patterns & practices*, 2008.
- [28] Moshkovich, H. M., Mechitov, A., “Verbal Decision Analysis: Foundations and Trends”, *Advances in Decision Sciences*, v. 2013, Article ID 697072, 9 pages, DOI: 10.1155/2013/697072, 2013.
- [29] Moshkovich, H. M., Mechitov, A. and Olson, D., “Ordinal Judgments in Multiattribute Decision Analysis”, *European Journal of Operational Research*, v. 137, n. 3, p. 625–641, 2002.
- [30] Pinheiro, P.R., Machado, T. C. S., Tamanini, I., “Verbal Decision Analysis Applied on the Choice of Educational Tools Prototypes: A Study Case Aiming at Making Computer Engineering Education Broadly Accessible”, *International Journal of Engineering Education*, v. 30, p. 585-595, 2014.
- [31] Ruano-Mayoral, M., Casado-Lumbreras, C., Garbarino-Alberti, H. and Misra, S., “Methodological framework for the allocation of work packages in global software development”, In *Journal of Software: Evolution and Process, J. Softw. Evol. and Proc.*, 2013.
- [32] Simão Filho, M, Pinheiro, P.R., Albuquerque, A.B., “Task Allocation Approaches in Distributed Agile Software Development: A Quasi-systematic Review”, In: *4th Computer Science On-line Conference 2015*, 2015, Zlín. *Proceedings of the 4th Computer Science On-line Conference 2015 (CSOC2015)*, Vol 3: *Software Engineering in Intelligent Systems*, 2015. v. 3. p. 243-252, 2015.
- [33] Simão Filho, M, Pinheiro, P.R., Albuquerque, A.B., “Task Allocation in Distributed Software Development aided by Verbal Decision Analysis”, In: *5th Computer Science On-line Conference 2016 (CSOC2015)*, Zlín. *Proceedings of the 5th Computer Science On-line Conference 2016*, 2016, in press.
- [34] Tamanini, I., “Hybrid Approaches of Verbal Decision Analysis Methods”, Doctor Thesis, Graduate Program in Applied Informatics, University of Fortaleza, 2014.
- [35] Tamanini, I., “Improving the ZAPROS Method Considering the Incomparability Cases”, Master Thesis, Master Program in Applied Computer Sciences, University of Fortaleza, 2010.
- [36] Tamanini, I., “Uma ferramenta Estruturada na Análise Verbal de Decisão Aplicando ZAPROS”, *Computer Sciences*, University of Fortaleza, 2007.
- [37] Tamanini, I, Carvalho, A.L., Castro, A.K.A., Pinheiro, P.R., “A novel multicriteria model applied to cashew chestnut industrialization process”, *Advances in Soft Computing* 58, 1, 243–252. 2009.
- [38] Tamanini, I., de Castro, A.K.A., Pinheiro, P.R., Pinheiro, M.C.D., “Towards an applied multicriteria model to the diagnosis of Alzheimer’s disease: a neuroimaging study case”, *2009 IEEE International Conference on Intelligent Computing and Intelligent Systems*, Vol. 3, pp. 652–656, 2009.
- [39] Tamanini, I., de Castro, A.K.A., Pinheiro, P.R., Pinheiro, M.C.D., “Verbal decision analysis applied on the optimization of Alzheimer’s disease diagnosis: a study case based on neuroimaging”, *Advances in Experimental Medicine and Biology* 696, 555–564, 2011.
- [40] Tamanini, I, Machado, T.C.S., Mendes, M.S., Carvalho, A.L., Furtado, M.E.S., Pinheiro, P.R., “A model for mobile television applications based on verbal decision analysis”, *Advances in Computer Innovations in Information Sciences and Engineering* 1, 1, 399–404, 2008.
- [41] Tamanini, I, Pinheiro, P.R., “Challenging the incomparability problem: an approach methodology based on ZAPROS”, *Modeling, Computation and Optimization in Information Systems and Management Sciences, Communications in Computer and Information Science*, 14, 338–347, 2008.
- [42] Tamanini, I; Pinheiro, P. R., “Applying a New Approach Methodology with ZAPROS”, In: *XL Brazilian Symposium on Operations Research*, p. 914-925, 2008.
- [43] Tamanini, I; Pinheiro, P. R. “Reducing Incomparability in Multicriteria Decision Analysis: An Extension of The ZAPROS Methods”, *Pesquisa Operacional (Print)*, v. 31, n. 2, p. 251-270, DOI: 10.1590/S0101-74382011000200004, 2011.

Effectively Testing of Timed Composite Systems using Test Case Prioritization*

Huu Nghia Nguyen
Montimage EURL
Paris, France

huunghia.nguyen@montimage.com

Fatiha Zaïdi
LRI-CNRS, Univ. Paris-Saclay,
91405, Orsay, France

fatiha.zaïdi@lri.fr

Ana R. Cavalli
SAMOVAR-CNRS, Télécom SudParis
Univ. Paris-Saclay, Évry, France

ana.cavalli@telecom-sudparis.eu

Abstract—A composite system consists of several components which can be developed separately and deployed in distributed environments. Executing test cases on such kind of systems requires more effort due to their size and their distributed environments. A critical issue is to prioritize efficient test cases to be firstly executed. We present in this paper a framework to generate test cases and to select the efficient ones to test the composite systems with taking into account time properties. Particularly, the framework generates a set of test cases based on a model of the system, which cover a given test objective. The test cases are then prioritized in an execution order to detect quickly faults, thus reducing the efforts of test execution and increasing the effectiveness of the testing process. The framework is complemented with an open-source toolchain for automating test case generation. It has been experimentally evaluated on the European Train Control System case study. The initial results show that the approach can save 40% of test execution effort.

I. INTRODUCTION

Even if formal methods can be used to avoid faults in the design and implementation processes, such as by generating code skeleton of system from its specification, testing remains the only means to gain some confidence in a final product [1]. The testing process consists of test case generation and test case execution. The model-based testing approach generates test cases based on formal models that represent the expected behaviors of System Under Tests (SUTs) rather than on source codes of the SUTs. Such testing process is known as black-box testing. An advantage of black-box test case generation is that it is possible to perform it as soon as a specification model is available, that is, before the production code is written.

The testing process is highly dependent on the faults detection ability of test cases. Test case generation is a main area of research in the field of software testing. Efficient test cases decrease the chances of failure of the system and ensure the quality of the system. They are very important in distributed testing of a composite system in which we need to execute them on the overall system. This is usually hard because tests must be deployed on a high number of components which can be autonomous, developed independently, and deployed in a (real) distributed environment. Metrics, *e.g.*, [2], used to evaluate test cases efficiency are usually based on their execution results such as, number of real faults detected, execution time, code coverage. This means that a test case can be known as efficient only after it is executed. The question is how we can predict a test case is better than another one in order to execute it firstly. Furthermore, as the set of faults in a SUT is usually unknown,

the definition of an efficient test case based on the number of real faults, which are detected by the test case, is not useful to practitioners who are creating test cases, nor to researchers who are creating and evaluating tools that generate test cases

The authors in [3] deal with distributed testing of systems that interact with their environment at physically distributed interfaces, called ports. The authors present several distributed conformance relations based on ioco, *e.g.*, dioco, c-dioco, p-dioco. An algorithm of test generation for each relation is also presented. The approach is based on a passive testing approach, *i.e.*, the testers do not send stimulus to the SUT but only observe it. The generation of test cases is not tackled in their framework. In [4], the authors use Timed Input Output Transition System (TIOTS) to theoretically reason about conformance, then they propose to use Timed Input Output Symbolic Transition System to describe test models. It gives an algorithm applying to offline testing to check observed logs of SUT against traces of the specification.

Optimization of test cases has been studied since long time [5]. Besides test cases prioritization approach for optimizing, there exists another approach, such as [6], [7], that tries to minimize the number of test cases in a test suite. This technique uses information about the program and the test suite to remove test cases which became redundant with time. An advantage of the prioritization with respect to this approach is that it does not discard or permanently remove some test cases from the test suite. Our framework can be used by both approaches as it gives the fault detection abilities of test cases. Indeed, one is able, based on these abilities, to decide, depending on the selected test strategy, either to execute firstly a test case, *e.g.*, when its ability is high; or to remove a test case, *e.g.*, when it can kill mutants that can be killed by another test case having higher ability.

We propose an effective test cases generation method applied to timed composite systems to maximize the ability of faults detection, consequently minimizing the effort, time and cost of tests execution. Our contributions are mainly on: (i) a formal model of timed distributed composite systems described by means of cooperating TIOTSs, (ii) a generation of distributed test cases, (iii) an evaluation of generated test cases to predict their abilities of fault detection, and (iv) the availability of an open-source toolchain¹ to support the automatic generation and evaluation of test cases.

The rest of the paper is structured as follows. Section II contains the basis of the proposed approach. Section III introduces an experimental evaluation of the proposed approach on the European Train Control System (ETCS). We also present

¹The work described in this paper has been partially financed by the ITEA3 MEASURE project n° 14009

¹ The tools are freely available under GPL 2.0 licence at <http://github.com/nhnghia/testgen-ixf>, and <https://github.com/nhnghia/iftree/2java>

in this section our tools' implementation for automatically generating and evaluating test cases and an evaluation of the scalability of the tools. We give the conclusion and future work in Section IV.

II. PROPOSED APPROACH

The SUT in our approach is defined by n components. Its testing system has n testers, each tester is attached to one component in order to test it. There is no need of communication between the testers. Each tester has a clock. These clocks progress at the same rate. A tester acts as the environment of its component, *e.g.*, it can send messages to the component and receive the responses from the component. It can also observe all inputs and outputs of its component with other components. A tester is put nearby enough with its component such that the communications between them have no delay, although the communications between components may have delays.

Let us take a simple example of a composite system having two components p^1 and p^2 . The component p^1 can output either a or c , denoted as $!a+!c$, while p^2 can receive either a or b , denoted as $?a+?b$. It is easy to see that their composition can do $!a$, then $?a$ and that p^1 (resp. p^2) cannot do $!c$ (resp. $?b$) in the composition. A test case tc of the system should be the sequence $!a;?a$. It is projected to get local test cases: a local test case of p^1 is $!a$ while the one of p^2 is $?a$. In the test case execution process, let say the tester of p^1 sees that p^1 emits a while the one of p^2 sees that p^2 receives b , *e.g.*, it is sent by the network. In that case, we will have two local verdicts: *pass* for the first tester, and *fail* for the second one. Hence the verdict of tc will be *fail*, thus a fault is detected. Let us note that the local test of p^2 cannot detect this fault if it is tested separately since $?b$ is allowed by the local model of p^2 .

Modeling. We use TIOTSs to model components of SUTs. Basically, a TIOTS is a Labeled Transition System (LTS) over the set of (real) events with inputs \mathcal{I} and outputs \mathcal{O} , and the set of time distances \mathcal{D} between events. Time distances between events are measured by duration variables $d \in \mathcal{D}$, *e.g.*, they are positive real numbers. A transition is *untimed* if it is labeled by an event α with $\alpha \in \mathcal{I} \cup \mathcal{O}$; or *timed* if it is labeled by a duration d with $d \in \mathcal{D}$. A *trace* of a TIOTS is defined as a sequence of labels $tc = \langle l_1, l_2, \dots, l_n \rangle$ with $l_i \in \mathcal{L}$ if there exist a sequence of transitions labeled by l_1, \dots, l_n respectively.

A SUT is a system consisting of several components. A component can interact with other components in the SUT but also with the environment that can be considered as a special participant of the SUT. Each component is identified by a unique name in its composite system. It has a model which describes the expected behaviors realized by the component. We model a component by a TIOTS. The basic events of a component are defined by a set of inputs or outputs of the component with the other participants in the composite system or with its environment.

Let p^1, p^2, \dots, p^n, e be a finite set of identifiers of components participating in the composite system to be tested, and $a, b, \dots \in \mathcal{M}$ be a finite set of messages, an *output* realized by the component p^1 to p^2 is denoted by $!a^{[1,2]}$ while $?b^{[2,1]}$ denotes an *input* b of p^1 from p^2 . In this definition, we use a special identity e to denote the *environment*, *e.g.*, $?a^{[e,1]}$ denotes

an input a of the component p^1 from the environment.

We also model *timed composite systems* by TIOTSs. The legal behaviors of a composition depends on the communication model used for the description of the message exchanges between its components. A communication model is characterized by a set of queues being put among components. A queue Δ is characterized by its size and its kind of message ordering, *e.g.*, ordered or unordered. We proposed 5 rules and their symmetrical rules [8] to construct a composition model from n local models.

Generating Test Cases. A (*timed*) *test case* is a sequence of inputs, outputs and durations. It represents a trace of a TIOTS and it covers some test objectives. We distinguish two kinds of test cases: global test cases and local ones. A *global test case* tc is a test case generated from a model of a composition system. On the contrary, a *local test case* tc^i of a component p^i participating in the system contains only events concerning the component. Once having a global test case, we need to project it on each participant component to get local test cases.

A *local test case* of tester t^k that tests the component p^k of the SUT is a sequence $tc^k = \langle d_1, \alpha_1, d_2, \alpha_2, \dots, \alpha_m \rangle$, where $d_i \in \mathcal{D}$ and each α_i is one of the following, where e denotes the environment:

- $!b^{[i,e]}$: an output of message b to the tester
- $?a^{[e,i]}$: an input of message a sent by the tester
- $!b^{[i,j]}$: an observation of a send message b of p^i to its component partner p^j , with $i \neq j$
- $?a^{[j,i]}$: an observation of a reception message a of p^i from its component partner p^j , with $i \neq j$

This definition requires that a test case always ends by a real event that could be an input or an output. Indeed, if we had a test case $\langle d_1, \alpha_1, \dots, \alpha_n, d_n \rangle$ with $d_n \neq 0$, then a tester would need to wait d_n time units before giving its final verdict. Thus this delay is not necessary as it has no effect to the verdict.

The *projection* process inputs a global test case tc and outputs a set of local test cases $\{tc^1, \dots, tc^n\}$. The local test case tc^k is obtained from tc by retaining only the events such that the component p^k participates in, *e.g.*, $!a^{[i,j]}$ or $?b^{[i,j]}$ with $k \in \{i, j\}$. The durations are preserved in all local test cases. Since an event, that does not concern p^k , will be removed in tc^k , there may be two durations that will be put successively in tc^k . We finally need to sum up all consecutive durations of local test cases.

A *verdict* of a local test case is given by comparing the expected result in the test case with the result given by the test execution. That is either an output, or a duration after that an output should happen. Let us take an example, a local tester needs to execute the local test case: $!a^{[e,1]}; 1; !b^{[1,2]}$. After the tester sends a to p^1 , there are the following possibilities:

- if the tester receives an output from p^1 then the verdict is *fail*,
- if the tester sees a message from p^1 to p^2 :
 - if the message is not b then the verdict is *fail*,
 - otherwise:
 - if the duration between the two messages equals 1 then the verdict is *pass*,
 - otherwise the verdict is *fail*

The execution of a global test case is done through executing its local test cases. A global test case tc gives a *pass* verdict only if all of its local test cases give *pass* verdicts. Basically, the test execution of a global test case will pass the following steps:

- 1) Generate local test cases $\{tc^1, \dots, tc^n\}$ from tc
- 2) Set the local test case tc^i to the local tester t^i
- 3) Launch each local tester
- 4) Wait until all testers finish
- 5) Emit a verdict *pass* if all testers emit *pass*, otherwise *fail*

Prioritizing Test Cases. An important factor to evaluate a good test case is the number of real faults detected by the test case. This means that the evaluation can be performed only after the execution of test cases. We propose to evaluate a test case by executing it against a simulator rather than a real implementation of the SUT. Thus it is evaluated even the SUT has not been yet completely implemented. This also permits to use fault injection testing [9] to evaluate the fault detection ability of the test case.

We use the mutation analysis technique to evaluate the prioritization of the generated test cases. Mutation analysis is a well-known approach to assess the quality of test cases or testing techniques [10]. We have built a simulator programming in Java for each component of a SUT. A simulator acts with respect to the model of its component. A SUT has n components that will be simulated by n simulators executing together. Artificial faults are then injected into the simulators. Each mutated version of a simulator, called *mutant*, is tested against test cases to detect deviations in behaviors of the original version that differ from the mutant, *i.e.*, the verdicts should be *fail*. We consider a mutant being killed by a test case if the verdict of executing the test case on the mutant is *fail*, *i.e.*, a fault is detected by the test case. A test case which does not kill (or detect) any mutants (or faults) is considered defective. The mutation score of a test case is measured by the percentage of mutants being killed by itself. Generally, a test case that has a higher mutation score is assumed to detect more real faults than the one that has a lower mutation score [11]. The evaluation of a local test case passes the following steps:

- 1) Generate a set of simulators $\{s^1, \dots, s^n\}$ from models of n components of the SUT,
- 2) Inject faults in each simulator s^i to obtain its mutants,
- 3) For each global test case tc :
 - a) Project to local test cases $\{tc^1, \dots, tc^n\}$,
 - b) Perform local test case tc^i on each mutant of s^i ,
 - c) Calculate the number of mutants being killed,
 - d) Set mutation score of the test case tc to the percent of mutants being killed.

We prioritize the generated test cases based on their mutation scores for the goal of fault detection rate, that is a measure of how quickly faults are detected during the testing process:

- 1) The first test case being selected is the one having the highest mutation score
- 2) The next test case is the one that kills the most mutants that are not killed by the previous test cases. This means that the second test case has the ability of detecting faults that cannot be detected by the first one
- 3) If two test cases tc_1 and tc_2 have the same mutation score, and the number of events of tc_1 is greater than the ones of

tc_2 , then tc_2 has a higher priority than tc_1 as its tester will have fewer interactions with the SUT but the same mutation score, *i.e.*, same ability of fault detection.

III. EXPERIMENTAL VALIDATION

Tool Implementation. We have built two open-source tools to support the framework, the TestGen-IFx to generate test cases, and IF2Java to generate Java simulator from Intermediate Format (IF) specification. IF is a formal language used to describe models of real-time systems. We use IF language to describe the models of each component of the SUT rather than using directly TIOTSs which are more intuitive to reason about the framework such as timed model, composition, trace, test cases. It allows to describe quickly a model due to its expressiveness: the IF models of the case study presented in this paper have 8 states and 14 transitions but their TIOTSs contain totally 49488 states and 80598 transitions. The readers are invited to refer to [12] for further details of IF language.

The TestGen-IFx tool has been developed to generate distributed test cases for distributed testing. It inputs an IF specification file, and a configuration file. The IF specification file contains all models of components of the SUT, each model is represented by an IF process. The selected exploration strategy is recorded into a configuration file. Depending on the kind of the strategy, some other parameters may be required, *e.g.*, search depth number, and test objectives. The main output of the tool is a set of local test cases. For each global test case found, the tool projects it in n files, each one contains a local test. The tool also displays statistics about the test generation process such as execution time, number of generated test cases.

The IF2Java tool has been developed for generating simulators that is used for the evaluation of the test cases generated by TestGen-IFx. It inputs an IF file, and outputs a Java file. The IF file contains models of components of a timed distributed system. The Java file contains several Java classes which each one describes the behavior of a component. We obtain a Java program, called simulator, after compiling the Java file and our simulator library. The simulator acts, *e.g.*, receiving inputs and sending outputs, *wrt.* its model, which is specified in the IF file. When executing, the simulator inputs a local test case, then gives a verdict.

Case Study Description. The ETCS is an automatic train control system designed to replace progressively the incompatible safety systems currently used by European railways. In the ETCS level 3 [13], a train is equipped with an Onboard Unit (OBU) and it is controlled by a Radio Block Center (RBC). We consider the ETCS as a composite system consisting of two distributed components, OBU and RBC, running in parallel, see Figure 1, and communicating by message exchanges via Global System for Mobile Communications - Railway.

In the ETCS, a train moves in virtual *moving blocks* defined by its RBC via Movement Authorities (MAs). A MA defines a location, called End Of Authority (EOA), to which the train is authorized to move. Beyond this location, the location is a danger point such as the entry point of an occupied block section or the position of the safe rear end of a precedent train. To ensure the train is able to stop at the given EOA, a MA contains also a *release speed* that is a speed limit under which the train is allowed to run. We do not present its formal models due to

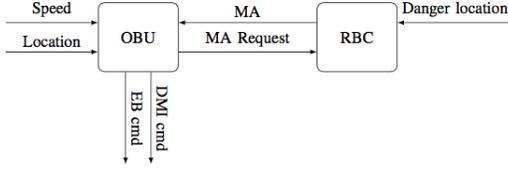


Fig. 1. Interaction between ETCS Components

lack of space². Particularly, the OBU receives its information about its current estimated location, *e.g.*, `ELocation`, then sends this location to RBC via MA requests. Based on received information: location of the train, and location of danger point, *e.g.*, `DLocation`, the RBC determines MAs of the train and sends it to the OBU. The release speeds are calculated by the procedure `getReleaseSpeed`. The calculation is based on the distance from the train to the EOA and on the capacity of the brake system of the train such that the train is able to stop the train at the EOA. By comparing the current estimated speed of the train, *e.g.*, `ESpeed`, to the release speed encapsulated in MA, the OBU may generate an emergency brake command, *e.g.*, `EBcmd(1)`, and Driver Machine Interface (DMI) commands to display relevant information to the driver, *e.g.*, `DMICmd(speed)`. The brake is applied until the train stopped [13].

To focus on release speed monitoring, we consider only release speed and distance to EOA in MAs. Since the MAs are needed to update periodically, the default period is 60s [13], we set one clock in OBU to issue a `MARequest`; and two clocks in RBC to send MA to OBU and to get `DLocation`. These clocks are reset when their events have occurred. For the purpose of experimental evaluation of the approach, we consider speed from 0 to 240 (km/h), location and distance from 0 to 300 (km), `EBcmd` from 0 (no brake) to 1 (brake) and `clock` from 0 to 60 (second).

Test Generation. A critical property of the ETCS to be tested is the capability of the system of taking over control if the driver appears to be going too fast. We need to consider a lot of scenarios to test this property such as when a break command is issued, when the current speed of the train passes over the limited speed, the break is released only if the speed is less than the limit speed, and so on. We present in the detail the test case generation for a specific scenario that represents the situation which caused the Spain train accident on 24 July 2013³. Considering that the train is in the indication state and is travelling in an area of track at 190 km/h meanwhile the speed limit is 80 km/h. At appropriate time-units, which are usually very close to each other, the RBC controls the train, checking the position, the speed and the acceleration. In this case of too high-speed, the OBU has to generate a brake command to reduce the speed, thus the accident would not happened because the driver cannot accelerate the train at 190 km/h. The test objective is formulated as the following, in which `{OBU}0` and `{RBC}0` are used to identify respectively the first instance of the OBU and RBC processes in the IF description:

² The complete IF description of the case study is available at <https://github.com/openETCS/validation/tree/master/VnVUserStories/VnVUserStoryMinesTelecom/05-Work/IF%20models>

³El Pais Journal, Saturday 27th of July 2013.

TABLE I. SCALABILITY OF TESTGEN-IFX

Depth	#Global Test Cases	Time(s)	Coverage of Variables		
			<i>v</i>	<i>l</i>	<i>x</i>
1	301	0.47		✓	
2	90,601	172.12		✓	✓
3	90,601	183.74		✓	✓
4	162,058	262.83		✓	✓
5	1,273,563	3803.81		✓	✓
6	65,913,084	90698.15	✓	✓	✓

$$tp_1 := \text{"process : instance = \{OBU\}0"} \wedge \text{"variable : m.speed = 80"} \\ \wedge \text{"state : source = INDICATION"} \wedge \text{"variable : v = 190"}$$

A global test case tc_1 being delivered by the TestGen-IFx tool is as the following:

```

?; e;      ELocation{100};    {OBU}0
!; {OBU}0; MARequest{100};  {RBC}0
?; e;      DLocation{103};   {RBC}0
delay 2
?; {RBC}0; MA{{30,80}};    {OBU}0
!; {OBU}0; DMICmd{80};     e
?; e;      ESpeed{190};     {OBU}0
!; {OBU}0; EBcmd{1};       e
  
```

The global test case is projected to obtain local test cases:

```

— Test case for OBU
!; e;      ELocation{100};    {OBU}0
!; {OBU}0; MARequest{100};  {RBC}0
delay 2
?; {RBC}0; MA{{30,80}};    {OBU}0
?; {OBU}0; DMICmd{80};     e
!; e;      ESpeed{190};     {OBU}0
?; {OBU}0; EBcmd{1};       e
— Test case for RBC
!; e;      DLocation{103};   {RBC}0
delay 2
?; {OBU}0; MARequest{100};  {RBC}0
!; {RBC}0; MA{{3,80}};     {OBU}0
  
```

Table I shows some metrics of test generations using Depth-First Search (DFS) exhaustive strategy to evaluate the scalability of TestGen-IFx. The experiments have been performed on a laptop with 2.2GHz Intel Core i7 processor and 16GB of RAM. The first column presents the depths explored. The second one relates to numbers of generated global test cases. The next one relates to the processing times in seconds. The three last columns represent the coverages of the variables in the models: train speed (v), train location (l), and danger point location (x). The rows corresponds to the results of different exploration depths. When the depth is 1, only one transition is fired, thus its events are executed: `?ELocation(l)`, `!MARequest(l)`. Since we consider data domain of variable l from 0 to 300, there are 301 possibilities of executions of the transition corresponding to 301 global test cases. These test cases cover (✓) the data domain of the variable location l using in the IF model of OBU. The number of test cases grows very quickly when the depth is 6. We obtain at this level a set of test cases that cover all possible values of the variables used in the models. Intuitively, we cannot execute all of these test cases against the SUT. We need to select the best test cases to be firstly executed.

Test Prioritization. After generating Java simulators representing the IF model of the ETCS system using IF2Java, we generated mutants by injecting faults that violate the safety properties of the ETCS. We use the Major [14] framework to generate the mutants. It allows us to create Java source code of mutants by injecting faults into some specific methods of a class. We inserted the 6 following types of faults:

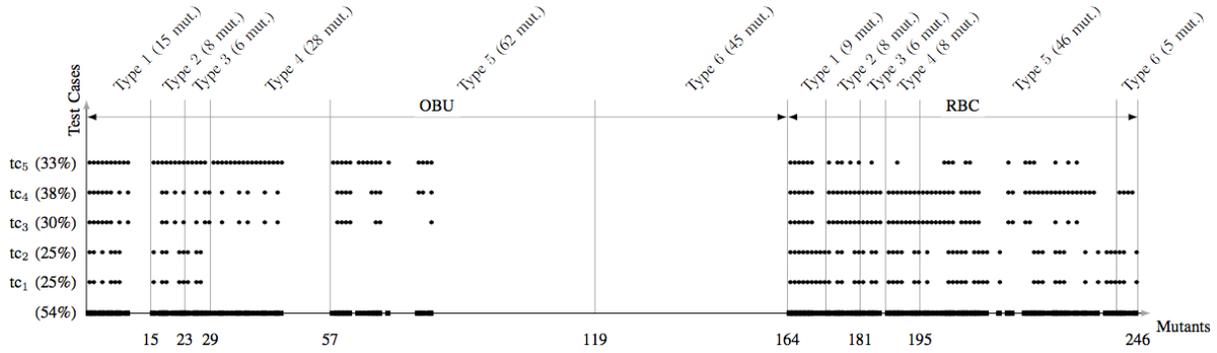


Fig. 2. Mutation Score and Mutant Coverage by Generated Test Cases

- 1) incorrectly implemented destinations of transitions,
- 2) incorrectly got inputs,
- 3) incorrectly sent outputs,
- 4) incorrectly evaluated guards of transitions,
- 5) incorrectly updated internal variables,
- 6) incorrectly updated clocks.

We obtain totally 246 mutants and it takes about 17 seconds. The fault *Type 5* generates the highest number of mutants (108: 62 for OBU and 46 for RBC). We then evaluate the generated mutants on 5 test cases, tc_1, \dots, tc_5 , which are generated from 5 scenarios to test the monitoring process of release speed⁴:

- 1) the current speed is 190 while the limit speed is 80,
- 2) the limit speed is 80 and the current speed of the train is 81, this just overs the limit speed,
- 3) the current speed is equal to the limit speed, 80,
- 4) when the break is hold,
- 5) when the break is released,

Figure 2 presents a cloud of mutants being killed by the test cases. *Ox* axis represents the mutants which are grouped by fault types, from 1 to 6, and by components which are either OBU or RBC. *Oy* axis represents the mutants killed by the 5 test cases. Average time of an execution process (compiling, executing and parsing results) of a test case on the 246 mutants takes about 240 seconds. Each mutant being killed by a test case is represented by a dot on the figure. For example, test case tc_1 kills 6 of 15 mutants, which are numbered 1,2,4,6,7,8 of fault *Type 1* of OBU. It kills in total 62 of the 246 mutants, thus its mutation score is 25%. Test cases tc_1 and tc_2 kill the same set of mutants, thus they have the same score. Test case tc_4 has the highest mutant score, 38%, then tc_5 with 33%.

After having the mutation scores, we prioritize the generated test cases with the goal of fast fault detection. Figure 3 represents the evolution of the Average Percentage of Faults Detected (APFD) [15] over the life of the execution of 2 test suites consisting of the 5 test cases on two prioritization ways. The values of APFD range from 0 to 100. A higher APFD number means faster (better) fault detection. Although the figure does not directly measure the fault detection ability of the two test suites, *e.g.*, they are always 54%, it allows us to compare the different prioritization orders to create faster detecting through the ordering of test cases. In particular, suppose we

place the test cases in order $tc_1, tc_2, tc_3, tc_4, tc_5$ to form a prioritized test suite T_1 . Figure 3(a) shows the percentage of detected faults versus the percentage of executed test cases of T_1 . After executing tc_1 , 62 of 246 faults are detected; thus 25% of the faults have been detected after executing 1 of the 5 test cases, hence 20%, in T_1 . No more new faults are detected after running test case tc_2 , thus 25% of the faults have been detected after 40% of T_1 has been used. The curve in the figure represents the cumulative percentage of faults detected. The gray area under the curve represents the APFD of the test suite T_1 , with 32%.

Figure 3(b) represents what happens when the order of test cases is changed to $tc_4, tc_5, tc_1, tc_3, tc_2$. It detects faults more quickly with APFD raising to 44%. It achieves the fault detection ability of T_1 , 54%, by using only 3 of 5 test cases, *i.e.*, the two other test cases detect the faults that were detected by the 3 previous test cases. Testers may not need to execute the two last test cases. Thus it can save 40% of test execution effort.

Discussion. Mutation scores of the generated test cases are low. Generally, this testing approach considers the SUT as a black-box. It does not know how the SUT works in back end. It focuses on the user perspective, *i.e.*, the inputs and outputs of the SUT. Consequently, it is not possible to guarantee that all mutations of the code in the SUT are covered as the test cases are generated without knowledge of the implementation of the SUT. This is known as a NP-hard problem [16]. Particularly, the test cases generated above tend to test a particular functionality of the ETCS system. Consequently, they are not able to detect violations of the other behaviors of the system.

The generation of mutants gives an overview of a distribution of fault types in a SUT as the number of generated mutants depends highly on the fault types being injected. Figure 4(a) presents a distribution of the 6 fault Types on the ETCS. The fault *Type 5* has the highest possibility of occurrence, with 44%. This is understandable because almost computation of the system is internal calculation. The fault *Type 6* has the second highest possibility of occurrence, that is 20%. It shows an important impact of correctly implementing the operations of clocks, such as update, reset, etc. in a timed composite system. Figure 4(b) presents the fault detection abilities of the 5 generated test cases on the 6 fault Types. It is clear that the fault Types 1, 2, 3, and 4 can be detected more easily than the others because they influence directly the outputs of the system.

⁴The complete formal description of the test objectives and their data can be found at <https://github.com/nhngnia/if2dot/tree/2java/example>

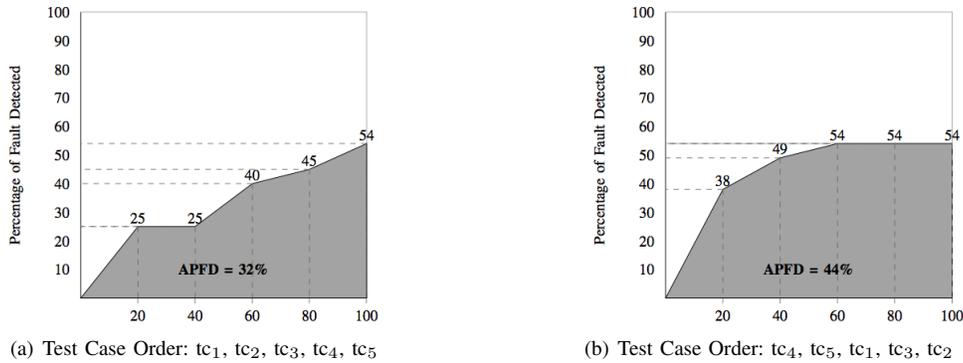


Fig. 3. Average Percentage of Faults Detected for Non-Prioritized (a) and Prioritized Test Cases (b)

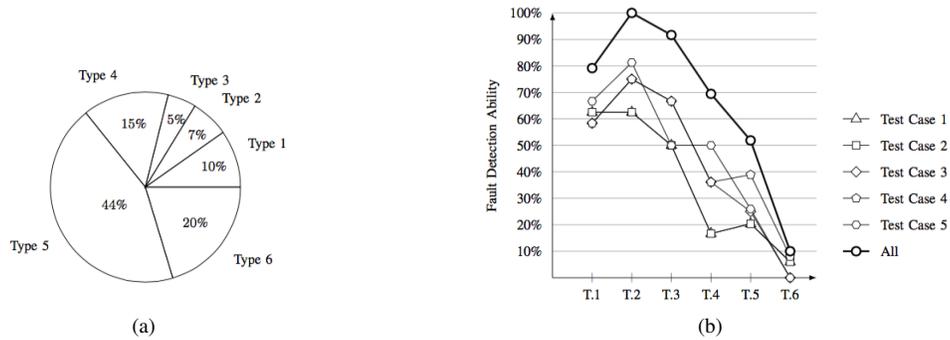


Fig. 4. Fault Types Distribution of the ETCS (a) and Detection Ability of the Generated Test Cases (b)

IV. CONCLUSION & FUTURE WORK

We have presented a framework for prioritization distributed testing of timed composite systems. The model of a SUT is composed of models of components described as TIOTSS. The testing system is established by several testers. Each tester tests one component of the SUT and there is no need of communications among them. Test cases are generated from the model of the SUT to cover some behaviors to be tested. We have performed an experimental validation of the approach on the ETCS case study. We also illustrated an example to how prioritize the generated test cases for faster detecting faults in the ETCS. The initial result shows that we can save 40% of test execution effort of the generated test cases. An open-source tool chain¹ has been implemented to automate the generation and evaluation processes of the test cases.

In future work, we firstly intend to improve the expressiveness of the test objective description to be able to specify more complex test scenarios. We also plan to consider the framework in case of unobservable communications among components of the SUT and in the case of clocks drifting.

REFERENCES

- [1] L. Cacciari and O. Rafiq, "Controllability and Observability in Distributed Testing," *Information and Software technology*, vol. 41, pp. 767–780, 1999.
- [2] R. Singh, "Test Case Generation for Object-Oriented Systems: A Review," in *Proc. of CSNT*, 2014, pp. 981–989.
- [3] R. M. Hierons, M. G. Merayo, and M. Núñez, "Implementation Relations and Test Generation for Systems with Distributed Interfaces," *Distributed Computing*, vol. 25, no. 1, pp. 35–62, Nov. 2011.
- [4] C. Gaston, R. M. Hierons, and P. L. Gall, "An Implementation Relation and Test Framework for Timed Distributed Systems," in *Proc. of ICTSS*, 2013, pp. 82–97.
- [5] W. E. Wong, J. R. Horgan, S. London, and A. P. Mathur, "Effect of Test Set Minimization on Fault Detection Effectiveness," in *Proc. of ICSE*, 1995, pp. 41–50.
- [6] N. Yevtushenko, A. Cavalli, and J. Lima, Luiz, "Test Suite Minimization for Testing in Context," in *Proc. of IWTC*, 1998, pp. 127–145.
- [7] N. Asoudeh and Y. Labiche, "Multi-Objective Construction of an Entire Adequate Test Suite for an EFSM," in *Proc. of ISSRE*, 2014, pp. 288–299.
- [8] H. N. Nguyen, F. Zaidi, and A. Cavalli, "A Framework for Distributed Testing of Timed Composite Systems," in *Proc. of APSEC*, 2014, pp. 47–54.
- [9] S. Ghosh, "Fault Injection Testing for Distributed Object Systems," in *Proc. of TOOLS*, 2001, pp. 276–285.
- [10] R. DeMillo, R. Lipton, and F. Sayward, "Hints on Test Data Selection: Help for the Practicing Programmer," *Computer*, vol. 11, no. 4, pp. 34–41, 1978.
- [11] R. Just, D. Jalali, L. Inozemtseva, M. Ernst, R. Holmes, and G. Fraser, "Are Mutants a Valid Substitute for Real Faults in Software Testing?" in *Proc. of FSE*, 2014, pp. 654–665.
- [12] M. Bozga, S. Graf, I. Ober, and J. Sifakis, "The IF toolset," in *Formal Methods for the Design of Real-Time Systems*, 2004, pp. 237–267.
- [13] UNISIG, "SUBSET-026 – System Requirements Specification," ERA, SRS 3.3.0, Mar. 2012.
- [14] R. Just, G. M. Kapfhammer, and F. Schweiggert, "Using Non-redundant Mutation Operators and Test Suite Prioritization to Achieve Efficient and Scalable Mutation Analysis," in *Proc. of ISSRE*, 2012, pp. 11–20.
- [15] G. Rothermel, R. H. Untch, C. Chu, and M. J. Harrold, "Test Case Prioritization: An Empirical Study," in *Proc. of ICSM*, 1999, pp. 1–10.
- [16] K. Chatterjee, Luca Alfaro, and R. Majumdar, "The Complexity of Coverage," in *Proc. of APLAS*, 2008, pp. 91 – 106.

Improving Accuracy of Patient Synthetic Data for Testing Medical Cyber-Physical Systems

Leonardo C. Santos, Lenardo C. Silva, Ana Luisa Medeiros, Hyggo Almeida and Angelo Perkusich
Federal University of Campina Grande, Campina Grande, Brazil
{leonardo, lenardo.silva, ana.medeiros, hyggo, perkusic}@embedded.ufcg.edu.br

Abstract

Medical Cyber-Physical Systems (MCPS) integrate the cyber space and physical world elements for promoting support for health assurance activities. MCPS are life-critical systems, demanding a strong engineering effort to guarantee safety, what directly impacts on testing process. Testing MCPS using real patients is very expensive and complex, since their lives are involved. Thus, the use of patient synthetic data becomes a promising approach. In this paper we propose a model for improving accuracy of patient synthetic data for testing MCPS based on regression models. We use an existing Patient Baseline Model to generate vital signs of patients, but improving the statistical analysis. Using our approach we increased in about 73.9% the quality of the regression models and, consequently, their accuracies.

Medical Cyber-Physical Systems; Statistical Analysis; Simulation; Testing; Patient Baseline Model (key words)

1. INTRODUCTION

The use of computing resources is increasing daily in personal and corporate environments. Since virtual entities directly react to stimuli produced by physical entities, these elements end up becoming a huge source of information for its users. This scenario, in which embedded computing units are in constant interaction with real world elements to monitor and control physical processes, forms the so-called *Cyber-Physical Systems (CPS)* [1].

CPS applied to health are commonly called *Medical Cyber-Physical Systems (MCPS)* [2]. In this sense, designing such systems has become an increasingly complex task due to the need to ensure the patient safety at runtime. This guarantee can be achieved through system verification and validation, what requires high abstraction level, realistic simulations and relevant tests.

Several models have been adopted as a way of representing the physical and cyber elements in the health field. Hotehama et al. [3] presented a cardiovascular model to

predict blood pressure and heart rate during physical exercises. Van Heusden et al. [4] proposed an artificial pancreas model for patients with type 1 diabetes mellitus, with the goal of improving glucose control in such patients. Wu et al. [5] and Bhaduri et al. [6] used artificial neural networks to investigate the correlation between blood pressure and some variables such as alcohol consumption, body mass index (BMI), age, and exercise, thereby building patient models to represent specific behaviors of the human body. Finally, Khan et al. [7] provided a glucose control system to be used to prevent hypoglycemic episodes, in which the patient model (i.e., the artificial pancreas) establishes a relationship between heart rate and blood glucose level.

Although there are several related works to develop MCPS, testing MCPS is still a challenge. Using real patients is very expensive and complex, since their lives are involved. Thus, the use of patient synthetic data becomes a promising approach. In this context, Silva et al.[8] presented a model-based architecture to support testing of MCPS. The authors introduced a Patient Baseline Model that uses regression models to generate synthetic data for the heart rate (HR), respiratory rate (RR), blood pressure (BP), and body temperature (BT) vital signs. In addition to proposing a new model, Silva et al. discussed the patient models proposed in other works, thus proving that the use of statistical data in order to obtain knowledge on human behavior is a common - but nontrivial - method. However, the potential predictor variables selected for the statistical analysis, as well as the use of the regression models for the vital variables, generated insignificant statistical results. In this case, the regression models inherent to the heart and respiratory rates, systolic blood pressure and body temperature represented only 48.9%, 31%, 51.1% and 48%, respectively, of the variability of the data contained in the samples selected for analysis.

Two main issues must be considered in the Patient Baseline Model: (i) predictor variables are not sufficient to explain the vital variables because the linear correlation among them is weak; (ii) the sample selected to perform the statistical analysis is heterogeneous, since the records

contained in this sample were collected in a time in which the individuals were admitted to intensive care units. This means that the patients were presenting the most varied critical health conditions.

In this research, we investigate the above mentioned issues to improve the quality of the prediction and accuracy of the regression models that compose the Patient Baseline Model. Therefore, in order to answer the following research questions, we declare their respective null hypothesis:

Q1: Can the regression models proposed by Silva et al. [8] be improved by modifying the predictor variables?

H1-0: There is no way to improve the regression models proposed by Silva et al. [8] by modifying the predictor variables.

Q2: Can the regression models proposed by Silva et al. [8] be improved by selecting a homogeneous sample?

H2-0: There is no way to improve the regression models proposed by Silva et al. [8] with the selection of another homogeneous sample from the database.

To investigate such issues, we present an experiment that was divided into the following three steps:

Step 1: Investigate the literature in order to identify the potential variables that are strongly correlated with each vital variable of interest (i.e., HR, RR, BP and BT);

Step 2: Select a sample from a database containing patient records in intermediate treatment periods in order to obtain better quality indicators for the regression models;

Step 3: Perform statistical analysis in order to obtain regression models that can better explain the vital variables of interest.

The remainder of this paper is organized as follows. Section 2 describes the characterization of the population of interest, as well as the statistical analysis performed to obtain the regression models for the vital variables. In Section 4, we discuss the threats to the experiment’s validity. Finally, in Section 5, we expose the final considerations.

2. MATERIALS AND METHODS

In the literature review, we identified some works that define a set of potential predictor variables for each vital sign. These variables are presented in Table 1. With the possibility of a multicollinearity problem in the construction of the regression models, only the systolic blood pressure was used, as safeguarded by Gavish [9], ignoring the diastolic blood pressure, as they have a strong correlation.

We used the same database used by Silva et al. [8] to collect patient’s records containing the larger number of potential predictor variables identified for obtaining the new regression models for vital signs. This database, so-called MIMIC II Clinical Database [13], is made available freely by the American service PhysioNet. The information in this database refers to patients admitted to Intensive Care Units (ICU) whose data were collected from bedside monitors and

Table 1: Potential predictor variables for each vital sign of interest, grouped by related work.

Vital sign	Predictor variables
BP [6]	Environment temperature, age, gender, body mass index (BMI), alcohol consumption, smoking, cholesterol and blood glucose.
BP [5]	Alcohol consumption, age and exercises.
BP and HR [3]	Weight, age, blood pressure at rest, heart rate at rest, exercise intensity, type of exercise and oxygen consumption.
All [10]	Age, gender, exercises, pregnancy, emotional state, hormones, medications, fever and hemorrhage.
BP	Age, exercises, stress, medications and diseases.
HR, RR and BT [11]	Age, exercises, stress, environment temperature, medications and diseases.
BP	Age, gender, environment temperature, emotional state, exercises, body position, medications, pain, recent meal, caffeine, smoking and bladder distention.
HR	Age, gender, exercises, emotional state, metabolism, fever and medications.
RR	Age, exercises, emotional state, fever and medications.
BT [12]	Age, environment temperature, emotional state, environment, exercises, patient’s normal body temperature and pregnancy.

hospital files. In addition to general patient data, other information can be found such as patient’s conditions at time of admission, vital signs and physiological parameters, drug administration, laboratory tests, and other information described in the doctor’s report.

From the analysis of the patients’ records found in the MIMIC II database, we identified and collected some of the variables presented in Table 1, such as age, gender, alcohol consumption, cholesterol, blood glucose, weight, height, oxygen consumption, medications and diseases, and the vital variables of respiratory and heart rate, blood pressure, and body temperature. These variables served as basis for the establishment of the linear regression models that compose the Patient Baseline Model.

2.1. Definition of the Population of Interest

Due to the large amount of records found in the MIMIC II database, as well as the possibility that some of the records contained errors or were duplicated, we identified the need of defining a population of patients for the study and a second sample for the validation of the proposed models. In the process of defining the population of interest, shown in Figure 1, we determined a set of rules to be ap-

plied, wherein the first rule was the “Specification of the population”. In this first rule, we selected the records of patients over the age of 15 years (because they have more stable vital signs), whose respiratory and heart rates, blood pressure, body temperature, glucose and CO_2 consumption were measured simultaneously.

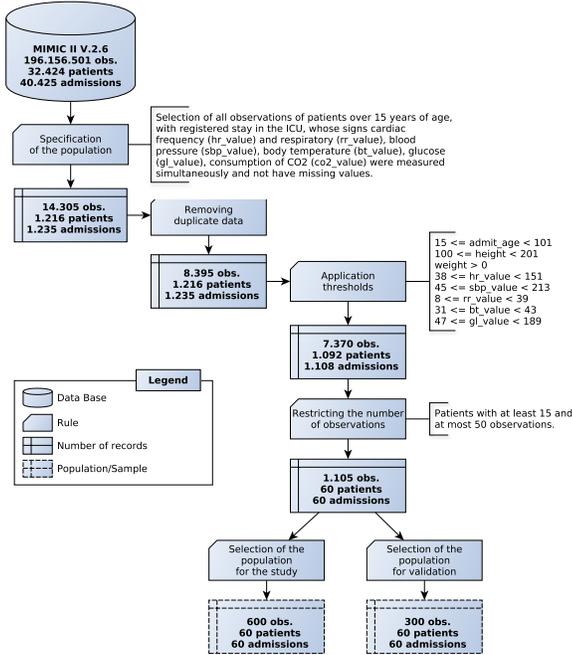


Figure 1: Defining the population of interest.

After the “Specification of the population”, we removed all of the duplicate data (“Removing duplicate data”) and then, in the “Application thresholds” rule, we defined the minimum and maximum values for the non-categorical variables with the purpose of removing data inconsistencies. As a reference for thresholds definition, we used a set of Clinical Guidelines described in [14, 15, 16, 17, 18]. With the “Restricting the number of observations” rule, we removed patient’s records who remained for a short (15 observations or less) or long time (50 observations or more) in the ICU. This allow us to select more stable patients.

The latest rules (“Selection of the population for the study” and “For validation”) are specifically related to the selection of two populations of interest for the study. At first, we selected randomly two records for each patient, resulting in 600 observations relating to 60 patients. The second population was defined to validate the regression models that were obtained from the first population. Thus, selecting randomly only one record for each patient results in 300 observations.

2.2. Statistical Analysis

This section describes the process of obtaining the regression models for the vital variables of interest which,

when interrelated, will provide the basic behavior of the Patient Baseline Model. The main statistical metric adopted in the evaluation process of the regression models was the square of the linear correlation coefficient between the answer variable (\hat{y}) and the adjusted values ($\hat{\mu}$), given by (1).

$$R^{2*} = cor(\hat{y}, \hat{\mu})^2 \quad (1)$$

In order to obtain the regression models, we used the generalized linear models (GLM) class. The method used to adjust the regression models was the Backwards Elimination [19] method. The reason for such choices is related to the large number of variables to be analyzed. In this method, the first linear regression is obtained with all of the potential predictor variables for each regression model, and then the variables are disregarded one by one according to the p -value¹ calculated by the t-test for significance testing.

In order to validate the obtained regression models, we used the following methods: (i) verifying the normality of the errors through the Shapiro-Wilk [20] normality test, which allows us to check if the model used is suitable for the data; (ii) comparison between the data obtained from the test sample and data generated by each vital sign regression model for this sample. In this comparison we performed the visual analysis of line graphs and the t-test for significance testing.

3. RESULTS

3.1. Regression Model for Respiratory Rate

As a starting point, we considered all of the variables present in the population of interest and the interactions between the variables $admit_age$, $height$, $weight$, rr_value , hr_value , sbp_value , bt_value , gl_value e $co2_value$, sex , due to the possibility that the interactions between them are significant.

The linear regression model used was the Normal Inverse given by (2) with canonical link function defined in (3). This is a particular case of the MLGs class [19] and was chosen since it best represents the sample. It is noteworthy that other models were tested.

$$\eta = \frac{1}{\mu^2} \Leftrightarrow \mu = \eta^{-\frac{1}{2}} \quad (2)$$

Where μ is the average of the respiratory rate (rr_value) variable, which we wish to model and

$$\eta = \beta_0 + \beta_1 X_1 + \dots + \beta_n X_n \quad (3)$$

is a systematic component, or linear predictor, in which β_0 represents the intercept coefficient, that is, when the value of all of the predictor variables of the model take the value 0, $\eta = \beta_0$.

¹According to Diez et al. [19], p -value is the variable used to obtain the test statistic that is equal or even more extreme than the one observed in a sample.

After the regression model for RR (MLG_RR) adjusted, the R^{2*} obtained for this model was 0.711. This means that MLG_RR explains 71.1% of the data variability contained in the vital variable rr_value . In practical terms, the MLG_RR representation, which represents the estimated average respiratory rate is given by (4)

$$\hat{\eta} = \beta_0 + \sum_{i=1}^{22} \beta_i X_i - \beta_{23} X_{11} X_{12} - \beta_{24} X_{12} X_{18} - \beta_{25} X_{19} X_{20} - \beta_{26} X_{11} X_{19} - \beta_{27} X_{15} X_{21} + \beta_{28} X_{15} X_{21} - \beta_{29} X_{15} X_{19} + \beta_{30} X_{20} X_{22} + \beta_{31} X_{12} X_{21} - \beta_{32} X_{21} X_{22} + \beta_{33} X_{19} X_{22} + \beta_{34} X_{15} X_{12} - \beta_{35} X_{19} X_{21} \quad (4)$$

where β_0 is the intercept, β_{1-35} are the coefficients inherent to each predictor variable, and X_{1-22} , a subset of the variables present in the population of interest. For more details see <https://github.com/leonardocsantoss/patient-baseline-model>.

Regarding the Shapiro-Wilk normality test used for determining the residuals present in the model, the p -value calculated was 0.995. This way, we can say with 95% confidence that the residuals present in MLG_RR have normal distribution and the model fits the data contained in the sample.

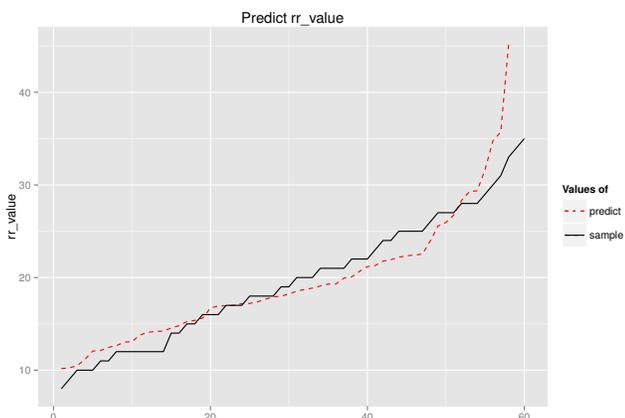


Figure 2: Comparison between the real data and the synthetic data calculated by MLG_RR for the rr_value variable.

In addition, to measure the accuracy of the MLG_RR , we compared the values of the rr_value variable in the test sample (real data), with the values calculated by the regression model for the same sample (synthetic data). In the comparative graph shown in Figure 2, it is possible to verify that the values calculated by the model (dashed line) are close to the real data (solid line).

The statistical evidence that these two data sets are equal is shown by the result of the t-test, in which the hypotheses are $H_0 : \beta_i = \beta_j$ and $H_1 : \beta_i \neq \beta_j$. Thus, to refute

H_0 (null hypothesis) implies that the two data sets are different. The result of the t-test for MLG_RR calculated a p -value = 0.5012. Thus, with 95% confidence, the null hypothesis was refuted, which leads us to conclude that statistically the two data sets are equal.

Once the process to obtain and fitting the regression models for vital signs was presented, we present only the results of the regression models for hr , sbp , and bt vital variables. Thus, we omitted the equations related to the linear predictor of these regression models.

3.2. Regression Model for Heart Rate

The regression model chosen for the heart rate variable (hr_value) was the Gama Linear Model with canonical link function given by (5). This regression model also belongs to the MLGs class and, when related to heart rate, is what best represents the variability of the data found in the sample. It is noteworthy that other models were tested.

$$\eta = \frac{1}{\mu} \Leftrightarrow \mu = \eta^{-1} \quad (5)$$

After adjustment of the regression model for HR (MLG_HR), we obtained the $R^{2*} = 0.822$.

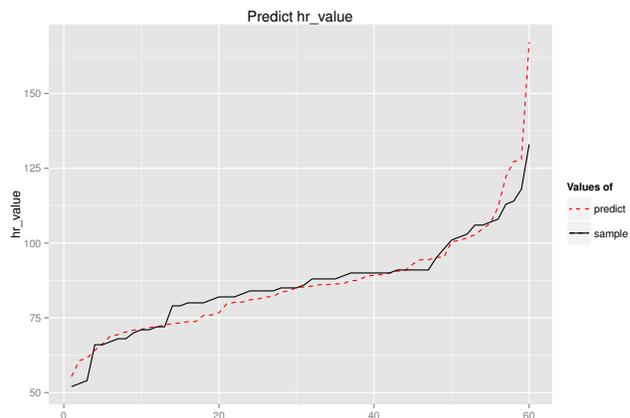


Figure 3: Comparison between the real data and the synthetic data calculated by MLG_HR for the hr_value variable.

The Shapiro-Wilk normality test used to verify the residuals in MLG_HR calculated a p -value = 0.9918. Thus, with 95% confidence, the residuals in these models also have normal distribution. Therefore, the MLG_HR is appropriate to the real sample data. Comparing the values calculated for this regression model with the variable hr_value values of the test sample (see Figure 3) using t-test for significance testing, we obtained a p -value = 0.7036. Statistically speaking, with 95% confidence, both data sets are equal.

3.3. Regression model for Systolic Blood Pressure

For the systolic blood pressure (sbp_value) variable, the Gama regression model was also used, whose canonical link

function was previously shown in (5). Other models have been tested, however, this was best represented the variability of the data. After adjusting the regression model for SBP (*MLG_SBP*), we obtained the $R^{2*} = 0.825$.

The Shapiro-Wilk normality test used to verify the residuals in *MLG_SBP*, calculated a $p - value = 0.06131$. Thus, with 95% confidence, the residuals in these models also have normal distribution. Therefore, the *MLG_SBP* fits to the real sample data. Comparing the calculated values for this regression model with the values of the *sbp_value* variable of the test sample (see Figure 4) using the t-test for significance testing, we obtained a $p - value = 0.3837$. Thus, with 95% confidence, the two data sets are statistically equal.

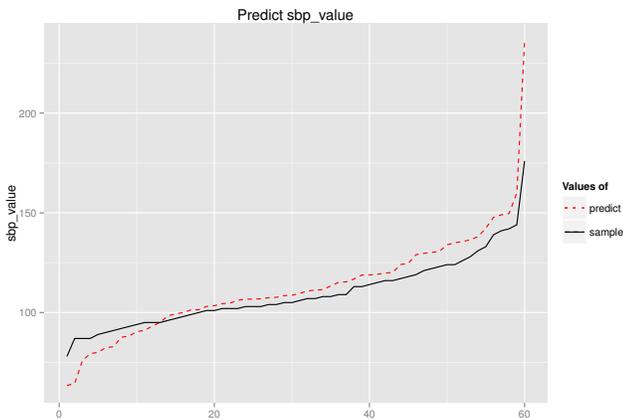


Figure 4: Comparison between the real data and the synthetic data calculated by *MLG_SBP* for the *sbp_value* variable.

3.4. Regression Model for Body Temperature

Finally, for the body temperature (*bt_value*) variable, we used the Gama regression model, whose canonical link function was presented previously in (5). After adjusting the regression model for BT (*MLG_BT*), we obtained the $R^{2*} = 0.755$.

The Shapiro-Wilk normality test used to verify the residuals in *MLG_BT* calculated a $p - value = 0.4675$. Thus, with 95% confidence, the residuals in these models also have a normal distribution. Therefore, the *MLG_BT* is suitable to the real sample data. Comparing the calculated values for this regression model with the values of the *bt_value* variable of the test sample (see Figure 5) using the t-test for significance testing, we obtained a $p - value = 0.3914$. Statistically speaking, with 95% confidence, the two data sets are equal.

4. DISCUSSION

In order to discuss the results obtained in this work, it is necessary to resume the two research questions defined in Section 1. According to the results obtained in the hypoth-

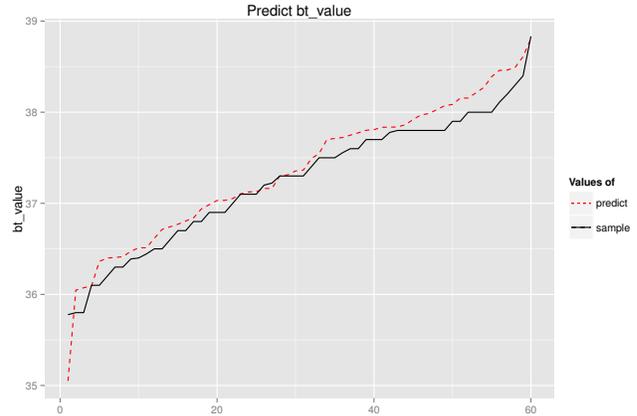


Figure 5: Comparison between the real data and the synthetic data calculated by *MLG_BT* for the *bt_value* variable.

esis tests related to their research questions, it was possible to refute the null hypothesis **H1-0** and **H2-0**, as shown in Table 2. This means that the use of a homogeneous sample from the MIMIC II clinical database, along with a set of predictor variables different from those proposed by Silva et al. [8], allowed us to obtain better regression models for the vital variables that make up the Patient Baseline Model. This was evidenced statistically through the values of the calculated R^{2*} metric for each regression model, as well as through visual analysis of real data compared to synthetic data generated by these models.

Table 2: Summary of the Results for the MLGs.

MLG	R^{2*}	Shapiro-Wilk test ($p-value$)	t-test ($p-value$)
MLG.RR	0.711	0.9950	0.5012
MLG.HR	0.822	0.9918	0.7036
MLG.SBP	0.825	0.0613	0.3837
MLG.BT	0.755	0.4675	0.3914

Regarding to threats to the validity of this study, we can take into account the following issue:

Restricted application domain: the used clinical database contains only data from intensive care units related to patients in critical health condition. This can interfere in the adjustment of regression models and the accuracy of the prediction of their vital signs. Therefore, this feature of the data sample restricts the application scope of the Patient Baseline Model;

Effectiveness of the Accuracy: some of the predictor variables shown in Table 1, such as exercise, environment temperature, stress, emotional state, pain, and so on, were not found in the clinical database used for this statistical analysis. Thus, these variables were excluded from the analysis, which can negatively impact in the accuracy of the regression models extracted for the vital signs considered in

the Patient Baseline Model. Furthermore, the number of variables used to generate the models was too high, which resulted in the increase of the model's complexity and, at the same time, raised its accuracy. Finally, it was not possible to determine the level of difficulty to use the model.

5. CONCLUSION

In this paper, we presented the Patient Model based model proposed by Silva et al. [8], which was built to serve as a basis to used in the MCPS validation and verification processes that requires interaction with real patients.

Initially, we discussed some related work that list the potential predictor variables for the respiratory and heart rate, blood pressure and body temperature vital signs. In addition, we presented a process to define the populations of interest for the study from a clinical database, including the data set used for statistical analysis and validation. Finally, we described the whole process to obtain and fit the regression models for the vital variables considered in the analysis. In the validation process of these models, we verified the normality of errors and compared the data generated by the regression models with the real values extracted from the test sample.

With the Patient Baseline Model representing the variability found in their vital variables (i.e., 71.1% of the respiratory rate, 82.2% of the heart rate, 82.5% of the systolic blood pressure, and 75.5% of the body temperature), we obtained regression models counting a gain of approximately 73%, when compared to the original model proposed by Silva et al. [8]. The results achieved in this study allow us to make more realistic simulations and generating relevant tests, increasing the confidence of such tests and minimizing the need to do clinical trials during the initial tests of a MCPS.

For future work, we intend to build a model representing the interaction between these regression models, allowing us to simulate different conditions regarding the basic health condition of a patient, characterized by the four main vital signs considered in this work.

References

- [1] M. Conti, S. K. Das, C. Bisdikian, M. Kumar, L. M. Ni, A. Passarella, G. Roussos, G. Tröster, G. Tsudik, and F. Zambonelli, "Looking ahead in pervasive computing: Challenges and opportunities in the era of cyber-physical convergence," *Pervasive and Mobile Computing*, vol. 8, no. 1, pp. 2–21, 2012.
- [2] I. Lee, O. Sokolsky, S. Chen, J. Hatcliff, E. Jee, B. Kim, A. King, M. Mullen-Fortino, S. Park, A. Roederer et al., "Challenges and research directions in medical cyber-physical systems," *Proceedings of the IEEE*, vol. 100, no. 1, pp. 75–90, 2012.
- [3] M. Hotehama, S. T. Lindsay, and T. Takemori, "Simulation of living behavior: a cardiovascular model for predicting blood pressure and heart rate," in *SICE 2003 Annual Conference (IEEE Cat. No. 03TH8734)*, 2003.
- [4] K. van Heusden, E. Dassau, H. C. Zisser, D. E. Seborg, and F. J. Doyle, "Control-relevant models for glucose control using a priori patient characteristics," *Biomedical Engineering, IEEE Transactions on*, vol. 59, no. 7, pp. 1839–1849, 2012.
- [5] T. H. Wu, G. K.-H. Pang, and E. W.-Y. Kwong, "Predicting systolic blood pressure using machine learning," in *Information and Automation for Sustainability (ICIAfS), 2014 7th International Conference on*. IEEE, 2014, pp. 1–6.
- [6] A. Bhaduri, A. Bhaduri, A. Bhaduri, and P. Mohapatra, "Blood pressure modeling using statistical and computational intelligence approaches," in *Advance Computing Conference, 2009. IACC 2009. IEEE International*. IEEE, 2009, pp. 1026–1030.
- [7] S. H. Khan, A. H. Khan, and Z. H. Khan, "Artificial pancreas coupled vital signs monitoring for improved patient safety," *Arabian Journal for Science and Engineering*, vol. 38, no. 11, pp. 3093–3102, 2013.
- [8] L. Silva, M. Perkusich, H. Almeida, A. Perkusich, M. Lima, and K. Gorgônio, "A baseline patient model to support testing of medical cyber-physical systems," *Studies in health technology and informatics*, vol. 216, pp. 549–553, 2015.
- [9] B. Gavish, I. Z. Ben-Dov, and M. Bursztyn, "Linear relationship between systolic and diastolic blood pressure monitored over 24 h: assessment and correlates," *Journal of hypertension*, vol. 26, no. 2, pp. 199–209, 2008.
- [10] J. V. González, O. A. V. Arenas, and V. V. González, "Vitals sign semiology: the new look to an actual problem," *Archivos de Medicina (Manizales)*, vol. 12, no. 2, pp. 221–240, 2012.
- [11] B. Vaughans, *Nursing Fundamentals DeMYSTiFieD: A Self-Teaching Guide*. McGraw Hill Professional, 2010.
- [12] K. Bonewit-West, S. Hunt, and E. Applegate, *Today's Medical Assistant: Clinical & Administrative Procedures*. Elsevier Health Sciences, 2014.
- [13] G. D. Clifford, D. J. Scott, and M. Villarroel, "User guide and documentation for the mimic ii database," *MIMIC-II database version*, vol. 2, 2009.
- [14] A. D. Association et al., "Diagnosis and classification of diabetes mellitus," *Diabetes care*, vol. 33, no. Supplement 1, pp. S62–S69, 2010.
- [15] A. V. Chobanian, G. L. Bakris, H. R. Black, W. C.ushman, L. A. Green, J. L. Izzo, D. W. Jones, B. J. Materson, S. Oparil, J. T. Wright et al., "Seventh report of the joint national committee on prevention, detection, evaluation, and treatment of high blood pressure," *Hypertension*, vol. 42, no. 6, pp. 1206–1252, 2003.
- [16] Y. Handelsman, J. Mechanick, L. Blonde, G. Grunberger, Z. Bloomgarden, G. Bray, S. Dagogo-Jack, J. Davidson, D. Einhorn, O. Ganda et al., "American association of clinical endocrinologists medical guidelines for clinical practice for developing a diabetes mellitus comprehensive care plan," *Endocrine Practice*, vol. 17, no. Supplement 2, pp. 1–53, 2011.
- [17] S. McGee, *Evidence-based physical diagnosis*. Elsevier Health Sciences, 2012.
- [18] U. D. of Health, H. Services et al., "The fourth report on the diagnosis, evaluation, and treatment of high blood pressure in children and adolescents. 2005," 2012.
- [19] D. M. Diez, C. D. Barr, and M. Cetinkaya-Rundel, *OpenIntro statistics*. CreateSpace independent publishing platform, 2012.
- [20] J. Royston, "An extension of shapiro and wilk's w test for normality to large samples," *Applied Statistics*, pp. 115–124, 1982.

Capture & Replay with Text-Based Reuse and Framework Agnosticism

Filipe Arruda, Augusto Sampaio and Flavia Barros

Centro de Informática

Universidade Federal de Pernambuco

Recife, Pernambuco, Brazil

{fmca, acas, fab}@cin.ufpe.br

Abstract—Software systems need to be constantly tested, either to verify changes or to check conformance to requirements. The current leading approaches to automate GUI tests are coding and the use of Capture & Replay (C&R) tools. Coding is usually associated with (even if *ad hoc*) reuse strategies, but requires from the developer specialized knowledge about the adopted framework. On the other hand, even though C&R is able to promote faster automation, it raises maintainability and scalability issues in the long term due to scripts scattering and rework for each new test case, because usually there is no associated reuse strategy. In order to combine the benefits of both approaches, we propose: an abstract and framework-free representation of test actions captured during testing activities; a text-based strategy that matches a new test case with previously recorded test actions; and a C&R tool that implements these concepts in the mobile context. We developed and evaluated our strategy in the context of a partnership with Motorola Mobility, achieving a reuse ratio up to 71% with time gains similar to traditional C&R approaches when compared to coding.

Keywords—test automation; capture and replay; reuse; mobile applications; natural language processing

I. INTRODUCTION

Despite the consolidation of testing as a verification activity, it is not always feasible to complete a testing campaign due to budget and time constraints [1]. Furthermore, when we consider the context of mobile devices, which is considerably different from Web and Desktop contexts, several other aspects should be observed, such as the wide range of sensors, distinct network providers and strong hardware dependency [2]. Thus, a large number of test cases (TCs) tends to be necessary to cover all these aspects, increasing the cost of the process.

Testing tasks tend to be repetitive. In Regression testing campaigns, for instance, every new version of the software must be tested, to verify whether it behaves as expected and to detect errors that may have been introduced in the modified code [3]. The simplest regression testing strategy consists of retesting the entire software by rerunning the reference test suite. However, due to cost constraints, alternative approaches select and run only a subset of the test suite [4]. In this context, test automation is seen as a way to reduce the time spent on testing activities [5][6], by mitigating the effort to manually execute an entire test suite. To automate test execution, Capture & Replay (C&R) tools are an alternative which does not demand programming skills and can be used during a testing campaign while tests are performed manually.

Regression testing arguably helps to control software quality, but new TCs must be continuously created to cover new features [7]. In some software systems, new features could have a high demanding rate, and automating the TCs for these features may be time consuming: 1) coding forces developers to acquire deep knowledge of the testing framework, design patterns etc. 2) current C&R tools require testers to execute the entire TC at least once. Besides, C&R approaches also suffer from high maintenance costs due to poor reuse, as noted, for instance, in the empirical assessment conducted in [8]. These evidences, associated to the use of an ambiguous language to specify requirements, affect negatively a direct mapping from test descriptions to scripts by automation tools, demanding testers to create a structured representation to enable an automatic and efficient transformation [9]. However, practical experiences in testing have shown that forcing programmers to adopt new notations is not the best option, reinforcing the use of well-known notations and environments [10][11].

Considering this scenario, we propose an abstract, recursive, text-based and framework-free representation, named *test action*, to store information that ranges from a simple test step to a complete TC or even a test suite. We apply text-processing algorithms to match new TC descriptions in natural language to previously recorded actions, reusing them to automate new TCs. We implemented these concepts into a C&R tool: *AutoMano*. Our tool was evaluated in an industrial context of a partnership with Motorola Mobility considering two metrics: reuse ratio and time spent to automate. As an important result we report a reuse ratio of 71% and a significant reduction in the overall implementation effort.

Section 2 discusses related work. Section 3 describes our proposed representation for test actions and our approach to reuse this representation by using string proximity and synonym matching techniques. Section 4 describes the tool and Section 5 details the conducted evaluation. Finally, Section 6 brings conclusions and future work.

II. RELATED WORK

Due to the vast literature on test automation, we focus here on approaches that are closely related to our work. UIAutomator¹ and Espresso² frameworks, developed by Google,

¹<http://developer.android.com/intl/en-us/tools/testing-support-library>

²<https://code.google.com/p/android-test-kit/wiki/Espresso>

allow test automation through coding based on atomic actions (e.g. `click.id(text1)`). However, the use of these frameworks demands specialized knowledge from the developer. Moreover, the automated TCs are framework dependent, and may become outdated when they use features that have been discontinued/deprecated after a framework or system update, requiring significant code refactoring. To minimize these effects, *AutoMano* proposes an intermediate representation and a central database that can be easily queried and updated.

Automation based on the C&R approach, on the other hand, does not require prior knowledge and is very fast. However, the created test scripts are typically linear, in the sense that they do not embody alternative paths, serving only the purpose of purely playback. Note that this characteristic restricts the reuse of actions to compose new TCs – for instance, just a slightly different disposal of the GUI may fail the execution of the test, as in the RERAN tool [12].

There are also hybrid approaches that mix the C&R approach with a strategy to capture keywords from the GUI to compose test actions. The test actions are stored as a script to be reproduced later. MonkeyTalk [13] and Robotium Recorder [14] are examples of frameworks that support C&R of keywords from applications built over Android or IOS platforms. Although they are well-consolidated commercial tools, they do not link test actions with TC descriptions (which is an important link we explore to ease the process of designing and reusing TCs). Moreover, even though we focus our comparison on C&R versus coding approaches for test automation, there are also other well-known strategies such as model-based testing (an example is MobiGUITAR [15]) and GUI ripping (see, for instance, [16]).

Our work was also inspired by an empirical analysis seen in [8], which shows that the development of test suites requires more time when programmable testing approaches are adopted (between 32% and 112%) compared to C&R approaches, however it is more time-saving over successive releases, because it is easier to maintain. Although these results were obtained in a different context (web), we assume that they could be similar or even more prominent in the mobile context. This happens because no reuse strategy are applied in C&R artifacts, while coded tests benefit from design patterns such as Page Objects. In our strategy we take advantage of the fast development observed in C&R tools without suffering maintenance issues raised from this approach by applying a reuse strategy.

Concerning TC generation from natural language requirements, NAT2TEST [17] presents a strategy that maps requirements written in natural language into TCs using a formal notation for requirements specification (SCR) as an intermediate formalism. Another example is presented in [18], in which requirements must be written in a more restrictive way as a strict if-then sentence template. Although these approaches aim to simplify the generation of TCs, these works: (1) severely restrict the input to a subset of a given natural language that must obey a particular grammar; (2) do not consider the reuse of test artifacts; (3) are not applicable to C&R approaches.

III. AUTOMATION STRATEGY

Our automation strategy is based on C&R, with focus on improving previous approaches particularly concerning potentializing reuse. Also, the input to our strategy is a TC written in free Natural Language (NL), particularly English; as far as we are aware, currently there is no strategy that automatically translates TCs written in (an unconstrained) NL into scripts of an automation framework, exploring reuse.

As previously discussed, the development of automated TCs, via C&R, usually results in platform dependent and hard-to-maintain code. For instance, considering the automation of a TC illustrated in Figure III, a direct mapping to scripts, besides the difficulty to find a connection between these representations, also hinders reuse possibilities. One could argue that, instead of mapping a TC to a single script, modularization could be explored by assigning a script to each step and reuse them in other TCs; but this is not sufficient to represent hierarchical steps. In addition, a direct mapping also makes scripts framework-dependent.

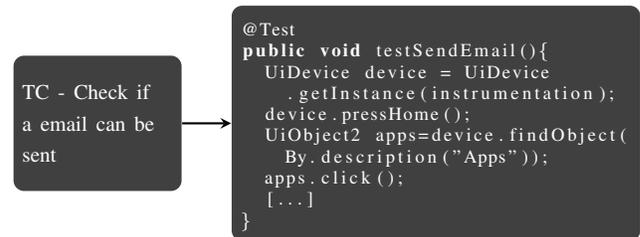


Fig. 1. Typical TC automation based on Capture & Replay

We propose the use of a middle-level representation, called *test action*, that fills the granularity gap between NL descriptions and GUI operations by supporting abstraction layers, composition and code-level interpretation. In this way, besides allowing actions to be retrieved, composed and reused by the test description itself, it is also possible to mitigate inconsistencies due to framework changes (deprecation, switched platform, business decision etc.), as detailed next.

A. Overall Architecture

In order to provide a better representation of the underlying abstractions of NL descriptions, test actions can be represented as recursive structures, inspired by the composite design pattern [19] illustrated in Figure 2 (a), supporting layers of abstractions that allow one to represent atomic operations, test steps, TCs or even test suites using the same structure.

It is worth mentioning that the high-level descriptions of atomic operations are automatically derived from screen interactions, while in composed test actions these NL descriptions are TC titles, step descriptions etc. Only atomic operations (that are predefined) are mapped to code-level scripts by using an interpreter to a specific framework (Figure 2 (b)), which can be dynamically instantiated using the factory method pattern [19]. To illustrate the framework agnosticism, we consider (in Figure 2 (b)) two frameworks: *UiAutomator* and *MonkeyTalk* with their respective interpreters.

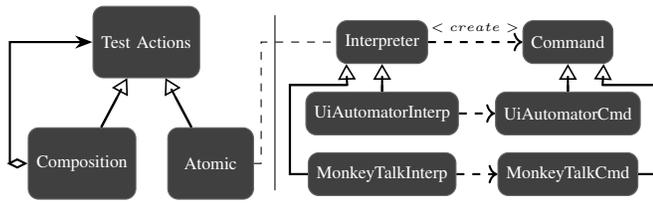


Fig. 2. Overall architecture

For instance, given the TC illustrated in Figure III to check whether an email can be sent, composed by several steps, the test action that represents this TC can now be structured as a composition of other test actions (each one representing a step), which in turn could also be composed by several screen interactions (represented as atomic test actions), as presented in Figure 3. We potentialise the reuse possibilities and provide the code script by only interpreting the atomic actions.

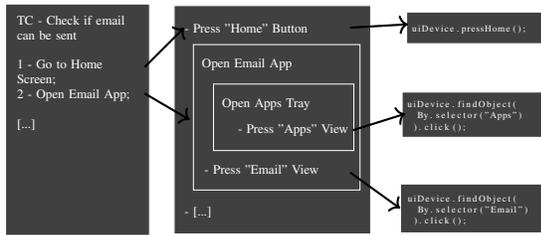


Fig. 3. Test case automation using hierarchical test actions

B. Reuse

In the beginning of an automation process, there are no previously created TCs, so there is no opportunity for reuse. However, as TCs are progressively automated, test actions are stored (typically in a database, as discussed in the next section) and can then be retrieved and reused in the automation of new TCs. Therefore, instead of capturing interactions all over again for every new TC, we employ an algorithm to match test steps written in English with the stored test actions, mitigating C&R issues regarding reuse, as noted in the empirical assessment conducted by [8]. These test actions may be organized and composed in any order to create others yet more complex ones.

The matching process is divided into three main operations: sentence tokenization; synonyms retrieval from the WordNet knowledge database [20]; and finally ranking the test actions using synonym equivalence and string proximity. The whole matching process is detailed in Algorithm 1. The function described as *CalculateSimilarity* is used to search test actions that match each step of TC description written in natural language. This function receives a test step and a description of a given test action as arguments, both written in English. Then, at lines 2 and 3, the sentences are split, also discarding some *stop words*. Then, beginning at line 5, each word of a test step (kwTS) is compared to each word of a test action (kwAD) and its synonyms retrieved from WordNet by verifying the Levenshtein distance [21]; for each successful matching (given

a threshold), the similarity level is increased. This function is executed for all stored action textual descriptions to find the best similarity level. Finally, if no test action found is similar enough (for a given threshold), the user must enter or create one specific test action for the test step being processed. For instance, if we consider a given TC step: "Compose a POP mail", the matching process is illustrated in Figure 4.

Algorithm 1 Calculating similarity using a knowledge-based approach

```

1: function CALCULATESIMILARITY(testStep, actionDescription)
2:   kwTS ← GETKEYWORDS(testStep)
3:   kwAD ← GETKEYWORDS(actionDescription)
4:   similarity ← 0
5:   for i ← 0..SIZE(kwTS) do
6:     synonyms ← GETSYNONYMS(kwTS[i])
7:     for j ← 0..SIZE(kwAD) do
8:       kwContains ← CONTAINS(synonyms[s], kwAD[j])
9:       kwProximity ← DISTANCE(synonyms[s], kwAD[j])
10:      if kwContains OR kwProximity < threshold then
11:        similarity ← similarity + 1/(COUNT(kwTS))
12:        break
13:      end if
14:    end for
15:  end for
16:  return similarity
17: end function
  
```

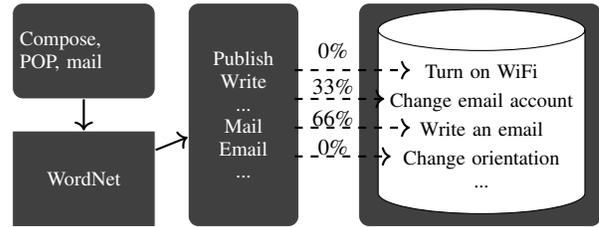


Fig. 4. Matching process

IV. IMPLEMENTATION

We implemented a tool, called AutoMano, which is able to capture user interactions on the phone and store them as test actions. It is worth noting that the application UI is web-based and this matching is transparent to the user. In summary, as in other C&R tools, a tester is able to automate an entire test suite without any programming skills during her common activities, reducing both time and effort to automate tests. Unlike other tools, however, it is possible to easily automate new TCs by just typing their descriptions (which potentially match actions previously recorded), favoring reuse because all actions are expressed as English sentences.

A. Capture & Replay

As observed in traditional C&R tools, AutoMano also captures and stores user inputs to reproduce them later. However, instead of capturing low-level events such as "clicking on (x,y)", our tool listens to the Android accessibility events³, which give us high-level descriptions of what was performed on the device. In this way we mitigate screen compatibility issues (due to different screen sizes), besides giving a more legible way to present information to the user. For instance, instead of "click on point (x,y)", our tool captures "click on

³<http://developer.android.com/intl/en-us/reference/android/view/accessibility/AccessibilityEvent.html>

button with description 'Apps'". Also, we created a custom keyboard that we install before capturing to get what the user is typing. Additionally, the user can create variables to reuse the same action in other situations, as shown in Figure 5.

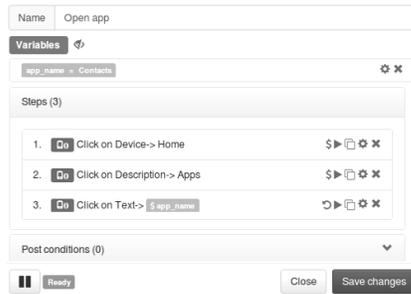


Fig. 5. Capture screen (AutoMano)

Then, after capturing a set of interactions, the user should give a representative description in English before storing them in a database (so these interactions could be retrieved by the algorithm presented in the previous section). The database used was Neo4j⁴ because it enables us to make graph queries which are useful to find equivalent subgraphs and analyze transitive connections among test actions. An example of how a test action would be represented is shown in Figure 6.

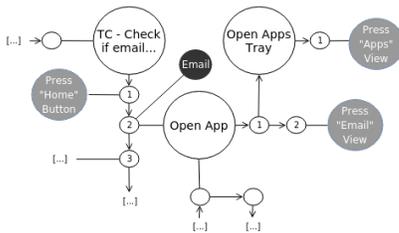


Fig. 6. How test actions are represented in Neo4j

Afterwards, to execute the test actions previously recorded, we first install an interpreter that reads test actions (JSON) and calls the corresponding framework (e.g., *UiAutomator*) methods. Then, the test action is retrieved from the database, converted to JSON and sent to device via socket communication. Subsequently, the interpreter translates each command of a test action, executes it and sends back the action results. Currently, our interpreter generates scripts for two frameworks: *UiAutomator* and a proprietary one. It is important to emphasize, however, that AutoMano can be easily tailored to use any other automation framework, since all commands are dynamically linked to scripts of a chosen framework. Finally, the results collected are shown on an HTML page to the user.

V. EVALUATION

In order to evaluate the effectiveness of our approach, we conducted an experiment in which we measured the time spent in test automation by developers coding scripts and that spent

by testers using AutoMano for C&R. In addition, we carried out a longitudinal analysis to investigate the evolution of the reuse ratio along a real-world automation task assigned to a tester in a project within Motorola Mobility. In this section we summarize the research questions, design choices, results and threats to validity.

A. Research Questions and Metrics

As a measurement mechanism, we adopted the GQM (Goal, Question, Metric) approach [22] to structure our evaluation in conceptual, operational and quantitative levels.

Study Main Goal: Verify whether our proposed C&R approach potentialise the reuse ratio of test actions during an automation task, while still being faster than coding.

[RQ1] Research Question 1: Is there indeed a reduction of the time spent to automate TCs via C&R with AutoMano when compared to coding?

Metric: Time spent to automate a set of TCs subtracting the execution and preparation time.

[RQ2] Research Question 2: Does our approach provide a satisfactory reuse ratio during automation?

Metric: Percentage of reused test actions: number of test steps retrieved from the database divided by the total number of test steps.

B. RQ1 - Design, Execution and Results

To obtain the metric associated with RQ1, we conducted an experiment that aimed to compare the time spent by developers to automate a given set of TCs with the time spent by testers to automate the same TCs via C&R, using AutoMano. These TCs were chosen according to the following criteria:

- They should be simple enough to mitigate the issue of different expertise levels of developers and testers.
- They should address recently added features, preferably not known by participants beforehand (to neutralize the testers prior knowledge).
- They must be real TCs, written by test designers under the same principles.

To conduct a precise computation of the time aspect, we prepared some guidelines and rules to the participants:

- Immediately before starting the automation activity, start the chronometer;
- Before asking for help, pause the chronometer and only resume it when the issue is solved;
- Pause the chronometer after finishing the automation of each TC, since the execution time to check if there are no errors should not be taken into account.

Regarding the selection of the participants, we chose 10 developers from an automation team working for Motorola Mobility whose experience ranged from preliminary to substantial. The project managers also assigned 10 testers with varied experience levels, who were trained for about 20 minutes on how to use the AutoMano tool. However, we ran the experiment with 7 available developers and 10 testers.

To ensure a proper execution, two assistants were assigned to supervise the experiment, enforcing the guidelines and

⁴<http://neo4j.com/>

assuring that the TCs were correctly automated. Whenever a problem was detected, the chronometer was stopped until the issue was fixed. Participants were not allowed to use functions or actions created prior to the current evaluation process. This way, TCs had to be automated from scratch. Additionally, it is worth mentioning that all participants used the same device and software version, and developers were instructed to use the same automation framework.

We protected the identity of the participants, to avoid any kind of retaliation due to performance indicators. As such, we assured to only share consolidated results.

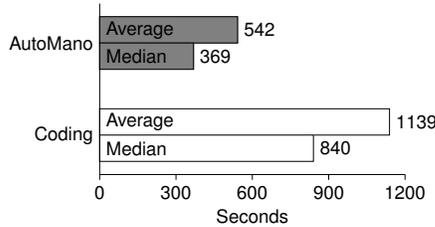


Fig. 7. Results: Comparing time to automate (AutoMano x Coding)

As shown in Figure 7, coding required an average of 101% more time to automate when compared to AutoMano, which actually is in accordance to the results achieved by Leotta et. al. (2013) [8], besides providing us a concrete evidence that AutoMano is faster than coding under these circumstances. To reach this outcome, we disregarded the values with more than 1.5 standard deviations from the mean, which excluded the highest automation time from both groups.

C. RQ2 - Design, Execution and Results

Concerning the RQ2, we prepared some artifacts to analyze the reuse ratio of test actions during a real task assignment in a Motorola Mobility project. For 3 weeks, we observed a novice tester using AutoMano to automate a set of TCs. Before we started to observe the tester, he was trained for 1 week on minor tasks using AutoMano, so that he could grasp the basic concepts related to testing activities, automation and the AutoMano itself. The aim was to provide a fairly meaningful analysis. The tester was asked to register each test step automated with AutoMano, informing if it was necessary to record the interactions or if there was a correspondent action already recorded in the database.

To track the reuse ratio evolution, we collected the percentage of reused actions over a time period. Then we observed how many actions were reused relative to the number of TCs automated. Figure 8 shows how reuse improved along the automation task. In general, as the number of TCs increased, the reuse ratio increased as well, reaching 71% with 31 TCs.

D. Threats to Validity

We discuss here the threats to validity of our evaluation.

Conclusion Validity: This work and evaluation were based on the assumption that it is reasonable to compare the time to automate TCs between two different groups: developers

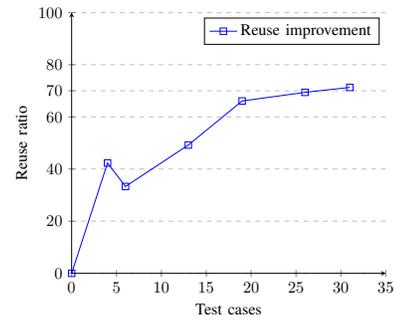


Fig. 8. Longitudinal analysis: Reuse ratio

and testers. We could have designed an experiment with only developers, but it would not have any practical or meaningful results since we seek alternatives to ease the burden of developers to automate and maintain a large set of TCs (by assigning them to testers, with no programming background, that could automate them even faster). Concerning the reuse strategy, we assumed that TCs have descriptions written in English and could be broken down into steps.

Internal Validity: Regarding the reuse ratio, the tester may have chosen only a subset of similar TCs, boosting the percentage of reused actions. If TCs from different products/projects are chosen, the reuse ratio might be distinct. However, we believe that as the number of TCs increase, we would notice a similar result. Concerning the experiment (RQ1), some participants could have measured a wrong time because they were responsible to register the time. For subsequent experiments, we intend to use tools that measure the time automatically based on key events. Also, because they were in working time and could have other appointments, some may have automated in a different pace if comparing to a real task.

External Validity: We conducted the analysis with artifacts and personal from a real project. However, products and TCs may vary from project to project, even the way to write TCs, which could affect our algorithm to reuse test actions. To mitigate this, we considered a traditional way to present TCs (description, steps and expected results) that we believe to be flexible enough to apply on most software projects. It is worth mentioning, however, that we only considered tests for the Android platform that could be automated by just screen interactions and were visually checkable. But the strategy proposed in this paper could be easily tested in other platforms (web, mobile) by extending the capture & replay modules, since the reuse strategy is framework-agnostic. Even so, regarding the time to automate, we got similar results to Leotta et. al. (2013) [8] experiment with web frameworks.

VI. CONCLUSIONS AND FUTURE WORK

Test automation based on coding scripts raises some issues: (1) it requires hiring specialized people to automate TCs; (2) code maintenance is not an easy task and it is often required, mainly because it is UI-based; (3) every new TC must be coded by a developer, even when it is similar to

previous ones. To deal with the two former issues, we proposed here a strategy and a tool that captures manual interactions made by testers, and converts them into an intermediate representation (test action) which is framework-independent and can be easily mapped into any automation framework. Only at runtime we interpret the framework script associated with atomic actions. Currently, the tool interprets scripts for *UiAutomator* and for a proprietary framework, and has been applied this to generate tests for verifying mobile devices in an industrial context of a partnership with Motorola Mobility. Because testers themselves can record their interactions, a dedicated automation team is not essential anymore; this eases maintenance because the testers only need to record new interactions on the device. To overcome the third issue, the tool is also capable of translating new TC descriptions into framework scripts by matching each test step with a test action previously saved in a database; as such, the more actions are stored, the better is the reuse ratio, consequently reducing the effort required to automate new TCs.

Unlike other capture & replay tools, AutoMano provides a straightforward way to reuse actions previously recorded by users, searching the database for actions whose associated descriptions are similar to the user input. The evaluation of this strategy brought evidences of a reuse ratio up to 71%, without drawbacks in the automation pace when compared to other capture & replay approaches.

Building upon the results already achieved, our work creates opportunity for several future research directions.

Matching improvements: To improve the matching success rate, we intend to adopt a controlled natural language to describe TCs, instead of using free English. We expect that a controlled grammar will gradually become representative enough to be the test action structure itself. Also, ontology-based representations for automation [23] could be used to express semantic relationships and context.

Capture and Replay improvements: As mobile devices frequently present new input sensors, it is important to cover these sensors by capturing interactions on new ones like voice or gesture motions. Regarding maintenance, we plan to adopt more flexible algorithms that are aware of UI changes, in order to handle minor UI changes.

Exploratory testing: An interesting topic for future investigation is the automatic generation of a large number of new TCs by combining test actions stored in the database, for the purpose of exploratory testing. This can be based both on a random strategy or on a more elaborate guided approach that only combines actions that are compatible. A notion of compatibility can be characterized by extending the structure of test actions with pre- and postconditions so that the sequential composition of two actions is meaningful only if the postcondition of the first action logically implies the precondition of the second.

ACKNOWLEDGMENT

This work is partially supported by CNPq (Grant 132329/2015-8) and Motorola Mobility. We thank Rodrigo

Folha and Cesar Albuquerque for help with implementation; Benicio Goulart, Alice Arashiro, Guilherme Almeida, Virgínia Viana and Dacio Mendonça for useful comments and follow-up; and for all testers who gave us feedback.

REFERENCES

- [1] I. Burnstein, *Practical software testing: a process-oriented approach*. Springer Science & Business Media, 2003.
- [2] R. Chandra, B. F. Karlsson, N. Lane, C.-J. M. Liang, S. Nath, J. Padhye, L. Ravindranath, and F. Zhao, "Towards scalable automated mobile app testing," Technical Report MSR-TR-2014-44, Tech. Rep., 2014.
- [3] H. K. Leung and L. White, "Insights into regression testing [software testing]," in *Software Maintenance, 1989., Proceedings., Conference on*. IEEE, 1989, pp. 60–69.
- [4] T. L. Graves, M. J. Harrold, J.-M. Kim, A. Porter, and G. Rothermel, "An empirical study of regression test selection techniques," *ACM Transactions on Software Engineering and Methodology (TOSEM)*, vol. 10, no. 2, pp. 184–208, 2001.
- [5] B. Beizer, *Software testing techniques*. Dreamtech Press, 2002.
- [6] C. Boyapati, S. Khurshid, and D. Marinov, "Korat: Automated testing based on java predicates," in *ACM SIGSOFT Software Engineering Notes*, vol. 27, no. 4. ACM, 2002, pp. 123–133.
- [7] C.-T. Lin, C.-D. Chen, C.-S. Tsai, and G. M. Kapfhammer, "History-based test case prioritization with software version awareness," in *Engineering of Complex Computer Systems (ICECCS), 2013 18th International Conference on*. IEEE, 2013, pp. 171–172.
- [8] M. Leotta, D. Clerissi, F. Ricca, and P. Tonella, "Capture-replay vs. programmable web testing: An empirical assessment during test case evolution," in *Reverse Engineering (WCRE), 2013 20th Working Conference on*, Oct 2013, pp. 272–281.
- [9] T. J. Ostrand and M. J. Balcer, "The category-partition method for specifying and generating functional tests," *Communications of the ACM*, vol. 31, no. 6, pp. 676–686, 1988.
- [10] A. Bertolino, "Software testing research: Achievements, challenges, dreams," in *2007 Future of Software Engineering*. IEEE Computer Society, 2007, pp. 85–103.
- [11] W. Grieskamp, "Multi-paradigmatic model-based testing," in *Formal Approaches to Software Testing and Runtime Verification*. Springer, 2006, pp. 1–19.
- [12] L. Gomez, I. Neamtiu, T. Azim, and T. Millstein, "Reran: Timing-and touch-sensitive record and replay for android," in *Software Engineering (ICSE), 2013 35th International Conference on*. IEEE, 2013, pp. 72–81.
- [13] CloudMonkey. (2013) Monkeytalk. [Online]. Available: <https://www.cloudmonkeymobile.com/monkeytalk>
- [14] Robotium. (2013) Robotium recorder. [Online]. Available: <http://http://robotium.com/>
- [15] D. Amalfitano, A. R. Fasolino, P. Tramontana, B. D. Ta, and A. M. Memon, "Mobiguitar: Automated model-based testing of mobile apps," *Software, IEEE*, vol. 32, no. 5, pp. 53–59, 2015.
- [16] A. Memon, I. Banerjee, and A. Nagarajan, "Gui ripping: Reverse engineering of graphical user interfaces for testing," in *null*. IEEE, 2003, p. 260.
- [17] G. Carvalho, D. Falcão, F. Barros, A. Sampaio, A. Mota, L. Motta, and M. Blackburn, "Nat2testscr: Test case generation from natural language requirements based on scr specifications," *Science of Computer Programming*, vol. 95, pp. 275–297, 2014.
- [18] M. Esser and P. Struss, "Obtaining models for test generation from natural-language-like functional specifications," *Proceedings of DX*, vol. 7, pp. 75–82, 2007.
- [19] E. Gamma, *Design patterns: elements of reusable object-oriented software*. Pearson Education India, 1995.
- [20] G. A. Miller, "Wordnet: a lexical database for english," *Communications of the ACM*, vol. 38, no. 11, pp. 39–41, 1995.
- [21] V. I. Levenshtein, "Binary codes capable of correcting deletions, insertions, and reversals," in *Soviet physics doklady*, vol. 10, no. 8, 1966, pp. 707–710.
- [22] V. Caldiera and H. D. Rombach, "The goal question metric approach," *Encyclopedia of software engineering*, vol. 2, no. 1994, pp. 528–532, 1994.
- [23] S. Paydar and M. Kahani, "Ontology-based web application testing," in *Novel Algorithms and Techniques in Telecommunications and Networking*. Springer, 2010, pp. 23–27.

Efficiently Measuring an Accurate and Generalized Clone Detection Precision using Clone Clustering

Jeffrey Svajlenko

Chanchal K. Roy

Department of Computer Science, University of Saskatchewan, Saskatoon, Canada

{jeff.svajlenko,chanchal.roy}@usask.ca

Abstract—An important measure of clone detection performance is precision. However, there has been a marked lack of research into methods of efficiently and accurately measuring the precision of a clone detection tool. Instead, tool authors simply validate a small random sample of the clones their tools detected in a subject software system. Since there could be many thousands of clones reported by the tool, such a small random sample cannot guarantee an accurate and generalized measure of the tool’s precision for all the varieties of clones that can occur in any arbitrary software system. In this paper, we propose a machine-learning based approach that can cluster similar clones together, and which can be used to maximize the variety of clones examined when measuring precision, while significantly reducing the biases a specific subject system has on the generality of the precision measured. Our technique reduces the efforts in measuring precision, while doubling the variety of clones validated and reducing biases that harm the generality of the measure by up to an order of magnitude. Our case study with the NiCad clone detector and the Java class library shows that our approach is effective in efficiently measuring an accurate and generalized precision of a subject clone detection tool.

I. INTRODUCTION

Clones are pairs of code fragments that are similar. Developers create clones when they reuse code using copy and paste, although clones may arise for a variety of other reasons [1]. Clone detection tools locate clones within or between software systems. Developers need to detect and manage their clones in order to maintain software quality, detect and prevent bugs, reduce development risks, and so on [1]. It is therefore important that the developers have high-quality clone detection tools, which requires knowledge of their detection performance.

Clone detection tools are evaluated in terms of recall and precision. Recall is the ratio of the clones within a software system a tool is able to detect, and precision is the ratio of the detected clones that are true positives, not false positives. While high-quality benchmarks have been proposed for measuring recall [2]–[4], accurately measuring precision remains difficult. In general, there has been a marked lack of research into methodologies for measuring a generalized precision both accurately and efficiently.

Precision is typically measured by validating a random sample of the clones the tool detects within a software system. For example, tool authors have checked on the order of 100 clones detected by their tool [5], [6]. However, this leads to a precision that is not generalizable and therefore not accurate. While a tool often detects a diverse variety of clones within

a software system, the detection report is often dominated by a few large groups of similar clones. These groups are distinct varieties of clone pairs that are common in the subject system, and are similar in terms of clone validation, but are not necessarily clones of each other (clone classes). A random sampling will mostly select from these similar clones, and a significant variety of clones are missed. This biases the measurement of precision to the varieties of clones that are most common in this subject system, but which may be rare in another software system. The result is a precision measurement that is not generalizable to another arbitrary software system, and is therefore not an accurate or useful measure for the users of the tool. While the variety of clones examined could be increased by sampling clones from a variety of software systems, this is difficult because clone validation is a very time-consuming process. Previous studies have had difficulty measuring precision for more than two [7] to eight [8] subject systems. The reliability of judges is also a major concern [9]–[11], so one cannot simply hire a large number of non-experts (e.g., undergraduate students) to scale the task. Random sampling is simply not an efficient way to select a variety of clones for measuring precision.

We propose a novel machine-learning-based approach for selecting a better sample of clones for measuring an accurate and generalized precision. This approach involves unsupervised clustering of the similar clone pairs detected in a software system (or collection of software systems) together. The goal is to cluster together the clone pairs that are similar in terms of the syntax and/or semantics that causes them to be validated as true or false positives, and which should be considered a distinct variety of clones when measuring precision. This is a finer-granularity concept than clone types. Precision can then be measured by validating a randomly chosen exemplar clone from each cluster. This efficiently maximizes the variety of the clone pairs examined when measuring precision. It also minimizes the bias caused by the particular distribution of the varieties of clones in the specific subject system, allowing a more generalized result to be obtained.

We evaluate our approach by clustering the clones detected by NiCad in the Java class library using K-means and measure precision. We compare this against the random sampling approach. Our approach doubles the varieties of clones examined within a sample size of 100 clones, while reducing biases in the measurement by up to an order of magnitude.

DOI reference number: 10.18293/SEKE2016-150

II. DEFINITIONS

Code Fragment: A continuous segment of source code, specified by the triple (l, s, e) , including the source file, l , the line the fragment starts on, s , and the line it ends on, e .

Clone Pair: A pair, (f_1, f_2) , of similar code fragments.

Clone Class: A set of similar code fragments. Specified by the tuple (f_1, f_2, \dots, f_n) . Each pair of distinct code fragments is a clone pair: (f_i, f_j) , $i, j \in 1..n$, $i \neq j$.

Type-1 Clone: Identical code fragments, except for differences in white space, layout and comments [1], [8].

Type-2 Clone: Identical code fragments, except for differences in identifier names and literal values, in addition to Type-1 clone differences [1], [8].

Type-3 Clone: Similar code fragments that differ at the statement level. The fragments have statements added, modified and/or removed with respect to each other, in addition to Type-1 and Type-2 clone differences [1], [8]. There is no agreement on the minimum similarity of a Type-3 clone [1].

III. CLUSTERING BACKGROUND

Clustering is an unsupervised learning technique for grouping similar objects. The goal is to group the objects such that an object is more similar to those within its own group, called a cluster, than those in other clusters. Given a dataset of n objects (data points), where each object is represented by a d -dimensional vector of measured features; the clustering problem is to label each data point with a value $[1, k]$, for a target number of clusters k . The data labels constitute a *clustering solution* to the data. Most algorithms require the data to be formatted in a n by d matrix of numerical values and for a number of clusters, k , to be specified.

A. Silhouette Metric

The silhouette metric [13] measures the quality of a clustering solution by how well each data point is clustered. The silhouette of data point i is shown in Eq. 1, where $a(i)$ is the average distance between data point i and the other points in its own cluster, and $b(i)$ is the lowest average distance between data point i and the data points in the other clusters. $a(i)$ measures how dissimilar the data point is to other members of its own cluster, while $b(i)$ measures how dissimilar the data point is to its most similar neighboring cluster. The silhouette of a data point ranges from -1.0 to 1.0. In this study we measure similarity and dissimilarity using the cosine similarity metric (Eq. 3).

$$s(i) = \frac{b(i) - a(i)}{\max\{a(i), b(i)\}} \quad (1)$$

A silhouette closer to 1.0 indicates that data point i is much more similar to the data points in its own cluster than those in its neighboring clusters ($a(i) \ll b(i)$), and is appropriately clustered. A silhouette closer to -1.0 indicates the data point is much more similar to data points in a neighboring cluster than those within its own cluster ($a(i) \gg b(i)$), and would be more appropriately placed in the neighboring cluster. A value closer to 0.0 indicates that the data point lies on the border of two clusters ($a(i) \sim b(i)$). The silhouette of a clustering

solution is measured as the average silhouette of its data points. This measures how well the data has been clustered, with a value closer to 1.0 being preferable.

The silhouette has a flaw that we must control for. If a cluster is created per unique data point, the silhouette trivially returns 1.0, a ‘‘perfect’’ clustering. Although such a clustering is unlikely to be useful. We want to cluster similar clones, not only identical clones. To control for this we also measure the *percentage of singleton clusters*. This is the ratio of the clusters that contain only a single unique data point (although they may contain multiple duplicate data points). We use this metric to determine if an increase in silhouette by adding additional clusters is due to a more natural clustering or due to an increase in singleton clusters.

B. Choosing a Number of Clusters

Often clustering is performed on data where the number of classifications in the data is unknown, as is the case with our clone data. A technique must be used to choose the number of clusters. In this paper we use the ‘elbow method’ [14] to estimate the natural number of clusters in the data. This involves plotting the quality (silhouette) of the clustering solutions as a function of the number of clusters, k , then looking for an ‘elbow’ in the plot where the gain in cluster silhouette by adding additional clusters suddenly and significantly drops. This estimates the natural number of clusters as the point where adding additional clusters no longer significantly improves the quality of the clustering solution.

C. K-Means Clustering with K-Means++ Initialization

K-means [15], [16] is an iterative clustering algorithm that aims to partition the data in a way that minimizes a loss function: the within-cluster sums of squares as shown in Eq. 2. Where k is the number of clusters, C_i is the set of data points in cluster i , $\vec{\mu}_i$ is the center of cluster i , and $d(\vec{x}, \vec{\mu}_i)$ is the distance between data point \vec{x} in cluster C_i and the cluster’s center $\vec{\mu}_i$. In document clustering, where documents are represented as weighted term-frequency vectors, the cosine distance, Eq. 3, is the preferred distance metric [17]. The algorithm is guaranteed to converge to a local optimum, although this may not be the global optimum.

$$\sum_{i=1}^k \sum_{x \in C_i} \|d(\vec{x}, \vec{\mu}_i)\|^2 \quad (2)$$

$$d(\vec{a}, \vec{b}) = 1 - \cos(\theta) = 1 - \frac{\vec{a} \cdot \vec{b}}{\|\vec{a}\| \|\vec{b}\|} \quad (3)$$

Given an initial set of cluster centers, the K-means algorithm iteratively updates the cluster centers to a local minimum of the loss function using the algorithm below. The clustering solution returned depends on the number of clusters and the initial cluster centers used.

- 1) Assign each data point to its nearest cluster center using the chosen distance measure.
- 2) Update the cluster centers as the mean of the points assigned to them.
- 3) Repeat steps 1-2 until the sums of squares converges or a maximum number of iterations has been reached.

We initialize the cluster centers using the K-means++ algorithm [18]. It aims to avoid poor clusterings by choosing random but evenly distributed cluster centers amongst the data. It guarantees a clustering solution that is at least $O(\log k)$ competitive with the optimal solution, and generally improves the speed and accuracy of K-means [18]. The algorithm chooses clustering centers using the following procedure, where D is the dataset:

- 1) Choose one $x \in D$ with uniform probability to be the first initial cluster center.
- 2) For each data point $x \in D$ the distance, $d(x)$, between x and its nearest previously chosen initial cluster center is measured.
- 3) Choose the next initial cluster center from $x \in D$ with a probability of choosing x of $\frac{d(x)^2}{\sum_{y \in D} d(y)^2}$.
- 4) Repeat steps 2-3 until k initial cluster centers have been chosen.

D. Principal Component Analysis (PCA)

Principle component analysis [19] is an orthogonal linear transformation of the data onto a new basis of linearly uncorrelated axes called principal components. These axes are chosen in decreasing order of the greatest data variance they explain. The dimensionality of the transformed data can be reduced by dropping the principal components that explain the least variance in the original data. Since the principal components are ordered by variance explained, only the Top- T principal components are kept for some target preservation of total variance explained.

IV. CLONE VARIETY

We define a **clone variety** as a collection of clones that are similar in terms of their clone validation for measuring precision. Two clone pairs are of the same clone variety if they share the same syntax, code-patterns and/or semantics that determine their validation as a true or false positive clone. This is different from clone type, which is a classification of clones based on the tool features (e.g. normalizations) needed to detect them (recall-focused). Clones of the same clone type can be very different, and be unrelated in how they are validated to measure precision.

As an example, clone detectors often report simple constructors as clones, that take a number of arguments and initialize member fields with their values. The clone detector reports similar pairs of these simple constructors as clones. These clone pairs are not necessarily clones of each other. They may vary by length, contents (e.g., names and number of parameters), and may be of different clone types. However, in all cases, validation depends on the decision of if two simple constructors form a true clone, so we consider these clones to be of the same variety of clone. Some other clone varieties observed in this study include: clones of methods that register action listeners, of auto-generated equals() methods, of methods implementing buffer slicing, and so on. While these clones can be found in many software systems, their frequency of occurrence depends on the subject system in question.

When measuring precision, we ideally only want to validate a single exemplar from each clone variety, as additional exemplars do not tell us anything new about the clone detector's precision. To measure a generalized precision, we want to give each variety of clone equal weighting, as different clone varieties may be more common or more rare in different subject systems. A generalized precision should be measured for a flat distribution of the different clone varieties found.

We do not attempt to build a taxonomy of all varieties of clones from the perspective of clone validation. Across the entire software development community, there is likely an unlimited number of clone varieties, and these varieties may overlap. Our interest is simply to judge if a cluster contains a single or multiple varieties of clones, and to measure the total number of clone varieties in a sample of detected clones.

V. MEASURING PRECISION WITH CLONE CLUSTERS

Here we discuss how precision can be measured using a clone clustering solution. The subject clone detector is executed for one or more subject software systems and the detected clone pairs are clustered. The goal is to produce a clustering solution where the clone pairs of the same clone variety are placed in the same cluster, and the clone pairs of different varieties are placed in different clusters. Precision is then measured by randomly selecting an exemplar clone pair from each cluster and manually validating them as true or false clones. The ratio of the exemplar clones judged as true clones by a clone expert is the precision of the tool.

This procedure maximizes the variety of the clones considered when measuring precision. It minimizes biases in the measured precision caused by the particular distribution of clone varieties in the particular subject systems under test, therefore measuring a more generalizable precision with respect to any arbitrary software system. It also minimizes the number of clone pairs that must be validated to achieve a desired variety of clones examined when measuring precision.

We cluster the clone pairs as a whole unit, not their individual code fragments. We cluster clone pairs rather than clone classes because clone detectors might mix true and false positives, and different varieties of clone pairs, within the same clone class. Clone classes can be unwieldy to validate, and there is no standard way of validating a whole clone class. Additionally, not all tools support clone class output, while all clone detection tools report clone pairs. We do not cluster clones of different clone types separately as it is not typical to consider them separately when measuring precision [1].

It is not realistic to expect a perfect clustering. The clustering solution might split clone varieties across multiple clusters, which can cause the clone varieties to have an uneven influence on the precision measured. The clustering solution may cluster multiple clone varieties into the same cluster, which may cause some clone varieties to be missed when choosing the exemplar clone pairs. Despite this, the goal is to achieve a better result than measuring precision by a pure random sampling of the detected clones. In our experiment, we find our approach achieves twice the clone variety in the same sample size,

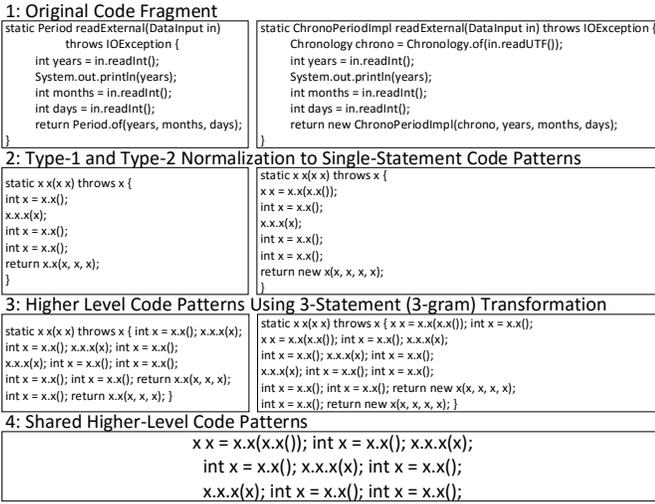


Fig. 1: Clone-Pair to Single-Document Conversion

while reducing biases that affect the generality of the measured precision by up to an order of magnitude.

VI. CLONE VECTORIZATION

Clustering algorithms require the data to be represented by a n by d matrix of numerical values, where n is the number of data points (clone pairs) and d is the number of features per data point. To convert the clones into this format we use standard document vectorization [17]. This treats a document as a bag-of-terms, and represents it as a vector of term-frequencies in term-space. The vector is weighted by the inverse-document-frequencies of the terms, under the assumption that frequent terms have less discriminating power. A document has the vector form shown in Eq. 4, where tf_i (term-frequency) is the frequency of term i in the document, df_i (document-frequency) is the number of documents term i appears in, n is the total number of documents, and d unique terms occur across all of the documents.

$$document = \left\{ tf_1 \log \frac{n}{df_1}, tf_2 \log \frac{n}{df_2}, \dots, tf_d \log \frac{n}{df_d} \right\} \quad (4)$$

Clones are pairs of similar code fragments (documents), so to apply document vectorization we need to convert the clone pairs into a single-document representation and choose the granularity of a term. From the definitions, the most important aspect of a clone is the code shared by its code fragments. The clone types indicate that the Type-1 and Type-2 differences in the clones should be ignored, so we normalize these differences in the code fragments, turning them into sequences of statement-level code patterns. However, it can be trivial for clones to share normalized code statements. Instead, we capture higher-level code patterns as sequences of multiple statements by applying an n -gram transformation over the normalized code statement patterns. Treating each code fragment as a bag of n -statement normalized code patterns, we extract the shared higher-level code patterns by computing the intersection of these bags, allowing duplicates. The single-document representation is then this bag of shared higher-level

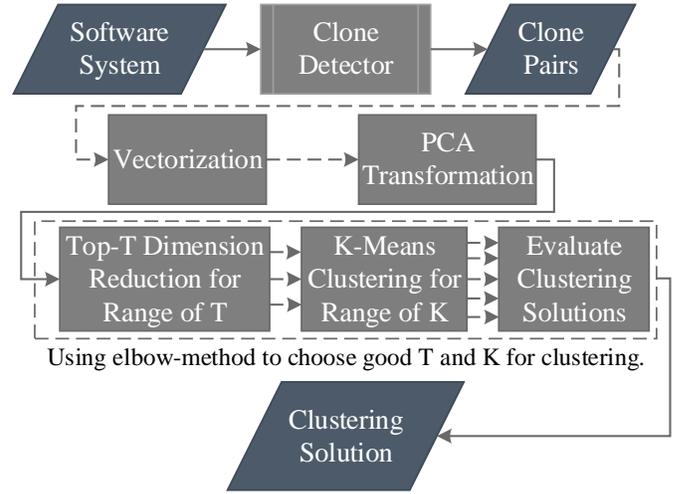


Fig. 2: Overview of Experimental Procedure

code patterns. An example of this is shown in Figure 1 for $n=3$. We drop the terms that appear in only a single clone pair as it discourages the clustering of similar clones, and singleton clusters are not desirable in our case. This single-document version is then vectorized as shown in Equation 4, with the n -statement code patterns as terms.

The goal is to cluster clone pairs of the same clone variety (Section IV), which we defined as the high-level shared syntax and semantics that determine the validation of the clone pair. We extract a clone pair's significant high-level cloned code patterns that define its clone variety. Then clustering is used to group it with clone pairs of the same clone variety, even if they are not clones of each other. The vectorization process is designed to only consider the significant cloned features of a clone pair, not the fine-grained cloned and non-cloned features that define variance within a clone variety.

VII. CLONE CLUSTERING PROCEDURE

In this section we describe the clone clustering procedure, which is summarized in Figure 2. For clustering, we use the K-means algorithm because of its availability in most data analysis frameworks. Our intention is for this methodology to be adopted by the cloning community, so it is important to use an algorithm which is accessible. In our experiment, we find K-means provides a good result for our use-case.

First, the clone detection tool is executed for the subject software system(s) and the detected clone pairs are collected. The clone pairs are then vectorized into d -dimensional vectors of real-numbers, as described in Section VI. The vectorized clones are of a very high dimensionality. Clustering algorithms are known to perform poorly in high dimensions [20]. In high dimensions, distance metrics begin to return similar values for any arbitrary pair of data vectors due to a high degree of orthogonality, which can cause poor clustering results with K-means. We perform dimensionality reduction using principal component analysis (PCA, Section III-D). First the clone vectors are rotated onto their principal components in order of decreasing variance explained, and then only the Top- T

principal components (dimensions) are kept for some target total variance explained. K-means (Section III-C) is executed on the transformed/reduced clone vectors to produce the clustering solution for a target number of clusters K . K-means is initialized using the K-means++ algorithm and performed using the cosine distance metric, which is the recommended metric for document-type data [17].

The final steps of the procedure require choosing T , the number of principal components to keep in dimensionality reduction, and K , the number of clusters. We have developed a procedure for choosing good T and K values based on the elbow-method (Section III-B) with the *silhouette* clustering quality metric (Section III-A).

First, multiple versions of the clone data vectors are produced for different Top- T dimensional reductions. For example, the Top- T principal components to preserve 50-90% (in increments of 10%) of the total data variance explained. For each Top- T , K-means is executed across a large range of K values, and for each clustering solution the silhouette is measured. The range of K should be chosen experimentally in order to observe the elbow in the plot of the silhouette versus the number of clusters. For each T considered, a plot of silhouette versus K is produced, and good values of T and K can be chosen from these plots considering the elbow method.

The value of T is chosen as the Top- T reduction that produces a stable plot of silhouette versus K with a well-defined elbow. We have observed that when too many principal components are kept, weak or multiple elbows are observed. When too few principal components are kept, the plot becomes unstable and no clear elbows are observed. So T is chosen as the reduction that produces the best plot with a clear elbow, and K is chosen as the number of clusters at the elbow. The clustering solution produced with the empirically chosen T and K parameters is used as the final clustering solution for the clones. While this procedure does not guarantee the best K and T values, and the silhouette metric does not understand the semantic concept of clone varieties, we find that it results in a good clustering solution for our use-case. It is not possible to manually inspect every clustering solution to pick the best K and T , and this procedure leads to a more natural clustering than choosing K and T arbitrarily.

VIII. EXPERIMENT

We test our approach by clustering the clones NiCad [21] detects in the top-level ‘java’ package of the Java 8 class library. We use the Java class library as it is a common target in clone studies [3], [8], and because it contains a wide variety of functionalities and therefore a wide variety of clones. We use the NiCad clone detector as it is a popular state of the art clone detector with strong recall for all three of the primary clone types [3], [22]. This ensures we are detecting a wide variety of clones for evaluating our approach.

We executed NiCad for the detection of function-granularity clones of 6 lines or greater with no more than 30% dissimilarity after blind identifier normalization and literal abstraction.

This is a generous configuration which ensures we detect a wide variety clones. In total, 14,076 clones pairs were detected. We vectorized the clones using a 3-statement (3-gram) representation. This is a compromise between capturing high-level code patterns, and being mindful that Type-3 clones contain statement-level gaps that may split sequences of shared code statements. The vectors are over 2711 term dimensions. This is a reduction from 7719 term dimensions after removing the singleton terms.

We use MatLab to perform principal component analysis on the clone data. This returns the vectors rotated onto a linearly uncorrelated basis of their principal components in decreasing order of the data variance explained by the principal components. Essentially 100% of the variance is explained by the first 1000 of the 2711 principal components, while 90% of the variance is explained by just the first 250 principal components.

We used the procedure described in Section VII to experimentally choose values of T and K . We explored values of T for preserving 50-90% of the total variance in the data. This corresponds to convenient T values of 15 (56%), 25 (63%), 50 (71%), 100 (80%) and 250 (90%). For each of these Top- T dimension reductions of the PCA clone vectors, we performed K-means clustering for a range of K from 20 to 1300. We executed K-means using MatLab with the cosine distance metric and initialized by the K-means++ algorithm, as previously described. We plot cluster silhouette versus K for each Top- T reduction in Figure 3.

The reduction to the first $T=100$ principal components (80% of total variance explained) appears to be the ideal reduction. It has a single well-defined elbow (natural clustering point), while the other reductions either have unclear or multiple elbows, or are otherwise unstable. It achieves a higher silhouette than keeping more dimensions (250), while having comparable silhouette to the further reductions (15-50).

We plot silhouette and percentage of singleton clusters versus K for K-means solutions with $T=100$ in Figure 4. The elbow method on the silhouette indicates that $K=100$ clusters is the natural number of clusters for this data. This is supported by the percentage of singleton clusters. Silhouette improves sharply up to 100 clusters, after which the gain in silhouette by adding additional clusters suddenly declines. As the silhouette slowly increases to 1.0 after 100 clusters, we see the percentage of singleton clusters increases linearly at a significant slope. The (weak) increase in silhouette after 100 clusters is most dominantly due to the increase of singleton clusters, which are not desirable. In contrast, before 100 clusters, the percentage of singleton clusters oscillates without any definite trend. The increase in silhouette as the number of clusters is increased to 100 is most dependent upon approaching a natural clustering point, not the number of singleton clusters.

Therefore our best K-means solution is achieved at $T=100$ principal components and $K=100$ clusters. To get our final clustering solution, we re-execute the K-means algorithm with 10 repetitions, meaning K-means is executed 10 times with different K-means++ initializations, and the best clustering,

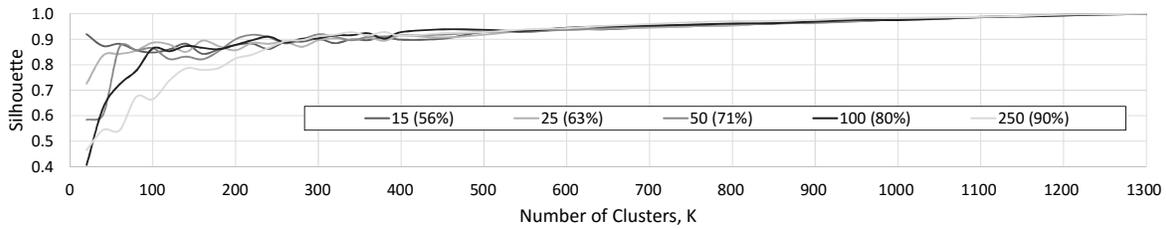


Fig. 3: Silhouette of K-Means Solutions Across k for Different Top-T Dimensionality Reductions

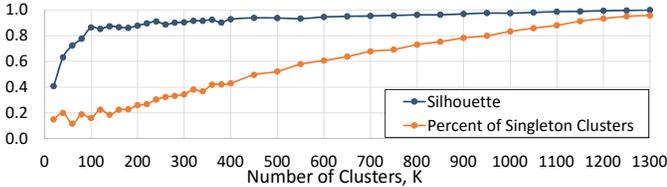


Fig. 4: Evaluation of K-means Clusterings for T=100

with the lowest total within-cluster sum of squares, is returned. This clustering solution has a silhouette of 0.86 with 16% singleton clusters.

IX. MANUAL INSPECTION OF THE CLUSTERS

We now manually inspect the clustering solution to see how well it isolates distinct clone varieties. We define a clone variety as a group of clones that are similar by the code patterns, syntax and/or semantics that determine their validation as true or false clones (Section IV). The goal was to produce a clustering of clones pairs where clones of the same clone variety are placed in the same cluster, and clones of different clone varieties are placed in different clusters. Although a perfect clustering was not expected, nor realistic to obtain. During this inspection we judged the clusters as strong or weak. A “strong” cluster is one that contains only a single variety of clones, while a “weak” cluster contains multiple varieties of clones. We also investigated if clone varieties were contained within a single strong cluster, or split across multiple. This manual inspection was done to judge the quality of the clustering, and is not a required component of the procedure for measuring precision (Section V).

When judging the clusters as strong or weak, we considered the syntax, semantics and code patterns of the clone pairs. In particular, how these elements affect the decision to validate the clones as true or false positives from the perspective of a variety of clones. A particular clone variety may span multiple clone types, and clones of the same variety may not be clones of each other. A few examples of the strong clusters we found include: clones of constructors that take a number of parameters and initialize fields with these values, clones of functions implementing buffer slicing on arrays of different primitive data types, and clones of short file-system operations wrapped in a common Java security manager code-pattern. In a few cases a large cluster of a single variety of clones contained a single or a small number of outliers. We judged these as strong clusters if the number of outliers was very small compared to the cluster size.

We judged 76 of the clusters as strong, meaning they contain only a single variety of clone. However, we noticed that some

of the strong clusters should ideally be merged. Specifically, multiple clusters contained clones of the same clone variety. This means that some of the exemplar clones chosen from the strong clusters will be of the same clone variety. The 76 strong clusters yield 56 distinct clone varieties when an exemplar clone is chosen from each of the strong clusters. While the splitting of a clone variety into multiple strong clusters is not a problem in terms of missing a variety of clones when measuring precision in this way, it does needlessly increase the number of exemplars to be examined, and adds a small bias in the overall precision measurement to these clone varieties. In this case it is minor, affecting only 11 of the 56 (20%) clone varieties found in the strong clusters.

We judged 24 of the clusters as weak, meaning they contain multiple varieties of clones. This does not mean the clones within the cluster do not share features, nor that they are all completely disparate. They contain clones that are sufficiently different to be considered different varieties of clones from the perspective of clone validation. When we investigated the weak clusters, we found that they could be converted into strong clusters if appropriately split. In most cases, a weak cluster was the merging of only 2-5 varieties of clones, while others had a more significant number of these hidden strong clusters. Some of these hidden strong clusters are of varieties of clones not previously seen, while some of them would ideally be merged with one of the existing strong clusters. Since some of the weak clusters contain clones not seen in the strong clusters, exemplars chosen from them still contribute to the variety of clones examined for measuring precision.

We randomly selected an exemplar from each of the weak clusters and found that 20 of these 24 clones were distinct from each other and not seen in the strong clusters. This suggests that some clone variety is missed when we only select a single exemplar from the weak clusters. This could be alleviated by selecting multiple exemplars from these clusters, although this is not necessary to measure a high-quality precision. A better use of additional efforts would be to extend the measure of precision to additional subject systems.

Overall, we judged 76 of the 100 clusters as strong and 24 as weak. In absolute number of clone pairs, 10,505 of the 14,069 clones pairs (75%) are in the strong clusters. The strong clusters perfectly capture the clone variety in the majority of the clone detection report, while the weak clusters still capture partial clone variety for the remaining 25%. Selecting one exemplar per cluster, a total sample size of 100 clones, yields 76 distinct varieties of clones: 56 from the strong clusters and 20 from the weak clusters. This is an efficiency of 76% in

terms of clone variety within the inspected clone pairs.

X. EVALUATION

We now compare our technique against the traditional pure random sampling in terms of clone variety and the biases that affect the accuracy and generality of the precision measure. As we found in the previous section, a sample of clones by choosing a single exemplar from each cluster in our clustering solution yields 76 distinct varieties of clones within a 100 clone sample (76% efficiency). For comparison, we selected 100 random clone pairs from the entire clone dataset (without clustering) and manually grouped them by clone varieties. A sample of 100 clone pairs by pure-random sampling yields only 37 distinct clone varieties (37% efficiency). This is the best-case we saw over several trials of selecting 100 clones at random. Our cluster-based approach achieves over twice the clone variety for the same clone validation efforts. We continued to randomly sample clone pairs until we reached parity. The random sampling approach required a sample size of 272 clone pairs to reach 76 distinct clone varieties at 28% efficiency. The efficiency in selecting a variety of clones by random sampling becomes less efficient as the sample size grows as it becomes more likely to choose varieties already seen. The random sampling approach requires almost three times the manual efforts to capture the same variety of clones. For this same effort, we could execute our cluster-based approach for additional subject systems and further increase the variety of clones considered.

While, with significant additional efforts, the traditional approach can match our approach in terms of total variety of clones considered, it cannot guarantee an accurate and generalized precision due to biases in the clone sample. We compare the distribution of the 76 clone varieties found by our cluster-based approach (100 exemplar clone pairs) and the traditional random sampling (272 randomly sampled clone pairs) in Figure 5. This plot shows the number of times each variety of clones appears in the clone samples produced by the two approaches. The clone varieties are ordered by increasing frequency, but the clone varieties do not necessarily correspond between the two techniques. For each approach, we highlight the first variety with a frequency greater than one.

For an accurate and generalized precision, each clone variety ideally has equal weighting in the measurement of precision. The clones considered by both techniques exhibit some bias due to some clone varieties appearing multiple times in the sample, causing these clone varieties to have a stronger weighting in the precision measurement. Our cluster-based technique has 13 clone varieties appearing multiple times in the sample, with most occurring twice, and a worst case of 5 exemplar clone pairs being of the same clone variety. The random sampling technique has 25 clone varieties appearing multiple times in its sample, with a range of re-occurrence of 2 to 64. With random sampling, five of the clone varieties have an order of magnitude higher impact on the precision measurement than the others, with a range of 12x to 64x in the worst case. These clone varieties significantly bias the

precision measurement. In contrast, our cluster-based approach reduces these biases by up to an order of magnitude. Overall, our approach exhibits very little bias.

While random sampling can estimate the precision of a clone detector for a specific subject system, it is not effective in measuring a generalized precision. Our cluster-based technique excels in measuring an accurate and generalized precision. Our technique requires a smaller sample size, therefore less efforts, than random sampling.

XI. MEASURING NiCAD'S PRECISION

To complete the case study, we also measure the precision of NiCad using our cluster-based procedure (Section V). We randomly selected one exemplar clone pair per cluster (100 clones) for manual validation. Since there is no clear and universally accepted definition of what constitutes a true clone, precision must be measured with respect to some context or use-case. In this case, we measured precision from a software maintenance perspective. We judged a clone pair as a true positive if its code fragments were not only syntactically and/or semantically similar, but if it would also be useful from a software maintenance perspective. Specifically, if one of the code fragments needed to be modified to fix a bug or to evolve the code, should the other code fragment also be considered for the same bug fix or code evolution. Otherwise we judged a clone to be a false positive. This is a rather strict criteria for a true positive clone.

Overall, we measured a precision of 74% for NiCad. The identified false positives were code fragments with similar syntax, but were not specifically relevant to software maintenance. In particular, clones of common Java programming and API idioms, as well as clones of auto-generated code. While these clones do share syntax, they are not useful for the identified clone-related software maintenance tasks. NiCad's precision is lower in this case as we purposefully used a generous configuration to ensure we detected a wide variety of both true and false clones to fully evaluate our cluster-based technique. Previous work [23], [24] has measured a precision of 80-90%+ when more conservative or recommended settings are used.

XII. THREATS TO VALIDITY

A different clustering algorithm might produce a better clustering. We use the K-means algorithm as it is simple and an implementation is available in most data analysis frameworks. This is important as our intention is for this technique to be adopted by the cloning community for measuring precision. We have found that K-means performed well in this case.

We limited our study to one clone detection tool and one subject system as the manual evaluation of the clustering solution and the clone validation efforts are very time intensive. We choose the subject system and tool very carefully to ensure we get representative results to the general case. The Java system library has a wide variety of functionalities from different domains and is therefore an ideal target to test clustering of a wide variety of clones. Of the available modern clone detection tools, NiCad has the best recall for the primary three

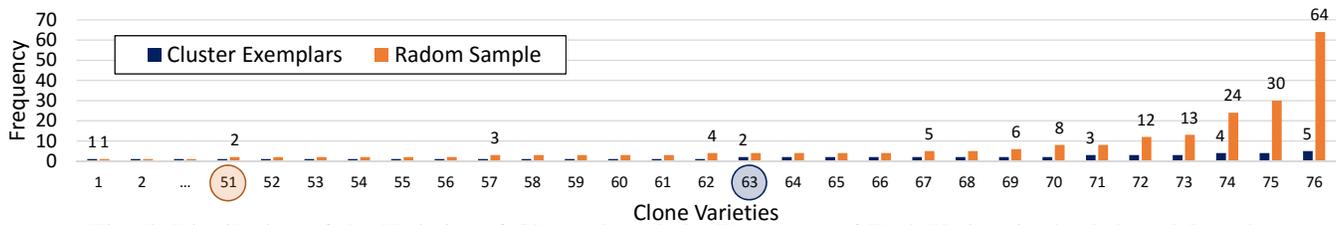


Fig. 5: Distribution of the Varieties of Clones Sampled - Frequency of Each Variety in the Selected Sample

clone types [22]. We configured it to be generous in detecting all types of clones to get the most wide variety of clones. The results should then represent any clone detection tool that targets the first three clone types.

As with any clone study involving manual clone inspection, the analysis, classification and validation of clones can be subjective. We took measures to reduce subjectivity and ensure fairness in the results. When comparing our cluster-based approach against traditional random clone sampling for measuring precision, we needed to classify the varieties of the clones, and measure the total variety of clones seen. We tracked our judgments on the clone varieties and the reasons for these decisions to ensure that the judgments were applied uniformly for the evaluation of both methodologies. Although there is always some subjectivity in manual clone analysis and classification, the comparison of the two approaches was kept fair by ensuring the classifications were applied uniformly.

XIII. RELATED WORK

Various benchmarks [2], [4], [8] have been created and experiments [3], [7], [8], [22] performed for measuring clone detection recall. Precision has been measured by manually validating all clones detected in a very small software system [25] or for a random sample of clones detected in one or more software systems [5]–[8]. Roy and Cordy [23] measured precision in a semi-automatic way specifically for synthetic clones injected into a software system. Yang et al. [26] used supervised learning to classify clones as per a manually labeled dataset. Ours is the first work to use unsupervised clustering to improve the efficiency and accuracy of measuring clone-detection precision.

XIV. CONCLUSION

We presented a novel approach approach for efficiently measuring an accurate and generalized clone detection precision. This involves clustering the detected clone pairs and validating an exemplar clone pair selected from each cluster. This ensures a wide variety of clones are considered when measuring precision with minimal bias due to the particular distribution of the varieties of clones within the subject system. We compared our approach against traditional random sampling. Our approach samples twice the variety of clones as the traditional approach for the same sample size, while reducing biases that harm the generality of the measured precision by up to an order of magnitude. The clones and clustering solution from this paper are available for interested readers¹.

¹www.jeff.svajlenko.com/seke16.html

REFERENCES

- [1] C. K. Roy and J. R. Cordy, "A survey on software clone detection research," Queen's University, Tech. Rep. 2007-541, 2007, 115 pp.
- [2] J. Svajlenko, J. F. Islam, I. Keivanloo, C. K. Roy, and M. M. Mia, "Towards a big data curated benchmark of inter-project code clones," in *ICSME*, 2014, pp. 476–480.
- [3] J. Svajlenko and C. K. Roy, "Evaluating modern clone detection tools," in *ICSME*, 2014, 10 pp.
- [4] J. Svajlenko, C. K. Roy, and J. R. Cordy, "A mutation analysis based benchmarking framework for clone detectors," in *IWSC*, 2013, pp. 8–9.
- [5] L. Jiang, G. Mishergchi, Z. Su, and S. Glondu, "Deckard: Scalable and accurate tree-based detection of code clones," in *ICSE*, 2007, p. 10.
- [6] Z. Li, S. Lu, S. Myagmar, and Y. Zhou, "Cp-miner: finding copy-paste and related bugs in large-scale software code," *Software Engineering, IEEE Transactions on*, vol. 32, no. 3, pp. 176–192, March 2006.
- [7] R. Falke, P. Frenzel, and R. Koschke, "Empirical evaluation of clone detection using syntax suffix trees," *Empirical Software Engineering*, vol. 13, no. 6, pp. 601–643, 2008.
- [8] S. Bellon, R. Koschke, G. Antoniol, J. Krinke, and E. Merlo, "Comparison and evaluation of clone detection tools," *Software Engineering, IEEE Transactions on*, vol. 33, no. 9, pp. 577–591, Sept 2007.
- [9] B. Baker, "Finding clones with dup: Analysis of an experiment," *Softw. Eng., IEEE Trans. on*, vol. 33, no. 9, pp. 608–621, 2007.
- [10] A. Charpentier, J.-R. Falleri, D. Lo, and L. Réveillère, "An empirical assessment of bellon's clone benchmark," in *EASE*, 2015.
- [11] A. Walenstein, N. Jyoti, J. Li, Y. Yang, and A. Lakhotia, "Problems creating task-relevant clone detection reference data," in *WCRE*, 2003.
- [12] MathWorks, "Silhouette plot - matlab silhouette," <http://www.mathworks.com/help/stats/silhouette.html>.
- [13] P. J. Rousseeuw, "Silhouettes: A graphical aid to the interpretation and validation of cluster analysis," *Journal of Computational and Applied Mathematics*, vol. 20, pp. 53 – 65, 1987.
- [14] DataScienceLab, "Finding the k in k-means," <http://datasciencelab.wordpress.com/2013/12/27/finding-the-k-in-k-means-clustering/>.
- [15] J. B. MacQueen, "Some methods for classification and analysis of multivariate observations," in *Berkeley Symposium on Mathematical Statistics and Probability*, 1967, pp. 281–297.
- [16] S. Lloyd, "Least squares quantization in pcm," *IEEE Trans. Inf. Theor.*, vol. 28, no. 2, pp. 129–137, 2006.
- [17] M. Steinbach, G. Karypis, and V. Kumar, "A comparison of document clustering techniques," in *KDD Workshop on Text Mining*, 2000.
- [18] D. Arthur and S. Vassilvitskii, "K-means++: The advantages of careful seeding," in *SODA*, 2007, pp. 1027–1035.
- [19] J. E. Jackson, *A User's Guide to Principal Components*, ser. Wiley Series in Probability and Statistics. Wiley-Interscience, 2004.
- [20] A. Rajaraman and J. D. Ullman, "Mining of massive datasets." New York, NY, USA: Cambridge University Press, 2011, ch. 7.
- [21] C. K. Roy and J. R. Cordy, "Nicad: Accurate detection of near-miss intentional clones using flexible pretty-printing and code normalization," in *ICPC*, 2008, pp. 172–181.
- [22] J. Svajlenko and C. Roy, "Evaluating clone detection tools with big-clonebench," in *ICSME*, 2015, pp. 131–140.
- [23] C. Roy and J. Cordy, "A mutation/injection-based automatic framework for evaluating code clone detection tools," in *ICSTW*, 2009, pp. 157–166.
- [24] H. Sajjani, V. Saini, J. Svajlenko, C. K. Roy, and C. V. Lopes, "Sourcererc: Scaling code clone detection to big code," in *ICSE'16*, Austin, Texas, 2016.
- [25] E. Burd and J. Bailey, "Evaluating clone detection tools for use during preventative maintenance," in *SCAM*, 2002, pp. 36–43.
- [26] J. Yang, K. Hotta, Y. Higo, H. Igaki, and S. Kusumoto, "Classification model for code clones based on machine learning," *Empirical Software Engineering*, vol. 20, no. 4, pp. 1095–1125, 2015.

NuSense: A Sensor-Based Framework for Ambient Awareness applied in Game Therapy Monitoring

Túlio H. Costa, Breno L. A. Paiva, Rinaldo V. Menezes, Lukas T. Ramos, Andrei G. Lopes, Frederico M. Bublitz

Center for Strategic Technologies in Health - NUTES

State University of Paraíba - UEPB

Campina Grande, Brazil

{tulio, breno.lacerda, rinaldo.vieira, lukas.teles, andrei.lopes, fredbublitz}@nutes.uepb.edu.br

Abstract — the usage of technologies for the treatment of patients is constantly emerging. A recent trend is the usage of games for healthcare, mainly in motor rehabilitation. The entertainment facet of the game attracts the attention of patients making them more engaged in the treatment. A difficult in the usage of commercial games is that they require full attention of the healthcare professionals, since they were not created to this purpose. To contour this kind of situation, decreasing the attention demanded by healthcare professionals, tailor-made games with monitoring capabilities have been developed. The problem with this approach is that these games falls short of entertainment, compromising their playfulness. Given this scenario, we believe that by adding monitoring capabilities to environments in which commercial games are used for health treatment is the best approach for game therapy. Therefore, it is necessary to create a mechanism that helps developers on building applications to monitoring patients' pre-determined characteristics during a gameplay. In this paper, we propose the NuSense, a sensor-based framework to support the development of applications to automatically monitoring people while they are playing electronic games as a health therapy.

Index Terms—Ambient Intelligence; games for health; monitoring capabilities; framework.

I. INTRODUCTION

The rise of new technologies has promoted the use of new approaches in healthcare, making changes in the traditional models of treatments. Some of these models, especially regard the patient motor rehabilitation, consists in the repetition of specific movements. This treatment is effective, but in the long run it tends to become unattractive to the patient, reducing the adhesion and efficacy of treatment [1]. In this case, the usage of electronic games has obtained good results, making this a good alternative due to the motivational and playfulness aspects [2].

The main aspects that contribute to these good results can be associated with the fact that this methodology can promote entertainment and consequently be used as a motivational tool - even if it is not the primary purpose, as in the case of Serious Games [3]. At this point, is important to highlight that usage of games goes beyond provide rehabilitation exercises [4], since they can be used for disease's prevention [5], and stimulate mental development [6].

The game engine improvements has allowed the use of increasingly realistic graphics. In addition, the enhancement of hardware has helped in the process of immersion and

improvement of user's experience by using image recognition, movement and sound techniques, such as the Microsoft Kinect Sensor (KS) [7]. This has allowed the use of commercial games, which focuses exclusively for entertainment, in the treatment of several diseases such as Parkinson's disease, Stroke, Huntington's disease, and Cerebral Palsy [8].

Many of the new technologies allows the monitoring of users by collecting information while they are playing electronic games [9]. Hence, health professionals can have a reduction in workload and devote their attention to important aspects of patient care. The automatic monitoring of people could help on identifying irregularities in therapies and optimize the work time of health professionals, which have his attention divided by several patients [10].

We cannot use commercial games to assess automatically the medical conditions of patients' treatment, because mostly of games do not approach health issues. Besides, the adaptation of such games is not always possible due to copyrights, interest of game companies and high costs involved. In the other hand, tailor-made games for health with monitoring capabilities has being developed, but they have a lack of playfulness, presenting a low quality and affecting the user's experience. The costs to produce a tailor-made game do not always worth [8].

To overcome this problem, we believe that the best solution is to add monitoring capabilities to environments in which commercial games are applied for game therapies. In this way, it is necessary to create a mechanism that helps developers on building applications to obtain and evaluate specific data and monitoring of patients during a gameplay. Also, the developers must be concerned mainly about the aspects of business logic - which are health issues.

In this sense, we propose a sensor-based framework to provide applications to monitoring of people in environments of game therapy and motor rehabilitation, aiming to reduce the overload of healthcare professionals. The framework will allow that the developers do not have to worry about non-trivial aspects of the technologies to build such applications, like the integration of different sensors, the data collection, treatment and transmission according with different sensors and techniques. At the same time, we intend to reduce the difficult to develop monitoring applications and accelerating the developing process.

II. RELATED WORKS

The emerging of new technologies has enabled the creation of games even more realistic. Due to its playful feature, games have been successfully applied in healthcare, mainly as a supplementary therapy for motor rehabilitation. Games are providing several benefits, including improvements in physical, sensory and cognitive functions, diagnosis, targeted treatments and remote monitoring.

A. Game Therapy

There are many works offering game therapies and having good results in the treatment of illnesses. In the case of the Brain Injury, it is the leading cause of death and disability worldwide [11]. That's why most part of game methodologies is focused on it [8].

For children with cognitive impairment, the use of games has been very restricted. However, we can highlight the JECRIPE game, designed for children with Down syndrome, focusing on the cognitive development of children of preschool age. With this game, the authors promoted activities such as imitation, perception, motor skills and verbal language [12].

As for people with advanced age, Kayama et al. [13] shows the Dual Task Tai Chi, a game focusing on dual activities, which showed an improvement in executive cognitive functions of patients. This kind of activity helps to reduce the fall risk in older people, who statistically have a higher probability of incidence [14].

The sedentary lifestyle and overweight have also been widely explored in the games for health area, especially with commercial games. With the convenience of playing in home, is being common for that public the use of technologies like KS, Wii balance board, Sony Eye Toy, mat of dance, and many others motion sensors.

To promote physical activity and reduce sedentary behavior in children, we can highlight O'Donovan et al. [15], which makes use of jogging and boxing games with the Nintendo Wii, and Maddison et al. [16] who used dance games for EyeToy. Both had positive effects on body composition of children.

However, games have been developed to utilize the same technologies of commercial ones. The competitiveness is currently explored as a motivating factor to overweight and sedentary persons. Zhang et al. [17] developed a game in a virtual network that simulates a bike marathon. They used sensors to catch some parameters of the users, thus, with help of adaptivity, the game was able to determine the user's performance based on their effort.

B. Monitoring Capabilities

For the monitoring of patients, there are games focusing on the treatment of overweight and sedentary people [18], as well as for the treatment of diabetes [19], in which therapists can monitor the performance of patients remotely and make recommendations. To monitoring of children while they're doing eye exam, the game PlayWithEyes [20] was made to do a pre-test and detect the children's visual acuity, so the ophthalmologist can monitor them while they're playing a ludic game which retains their attention.

Konstantinidis et al. [21] developed an application for monitoring older people with the integration of several sensors, like KS, Wii Balance Board and Android smartwatch. They used standard web technologies to encapsulate the data in the cloud, so the applications are able to work in a single data format with different sensors.

Apiletti et al. [22] developed a framework for monitoring of patients that allow a real-time analysis of their physiological data. With a detailed analysis, a risk function is calculated, and the current state of the patient is provided. In the field of Body Sensor Networks (BSN), Iyengar et al. [23] proposed a framework to assist in the development of applications for monitoring of patients. This framework was intended to be used in postural recognition applications and physiotherapy.

Pirovano et al. [24] created a game engine called IGAR, which focus on the motor rehabilitation of patients at home. The IGAR engine provides a real-time correction of user's mistakes made at therapies. One of the features of the IGAR engine is a middleware layer that allows the use of several sensors into the game to be developed. However, this approach is focused on building tailor-made games, which brings us back to the initial question – tailor-made games may affect the user experience.

Yet, the complexity of the development of monitoring applications still large. Besides, few solutions allow the integration of different sensors, the flexibility on generating outputs to other devices or technologies, and the information storage about the monitored environment. Lastly, still the developers' assignment to abstract the complexity of the sensor, when they could be focused on the application domain, such as the variables to be monitored.

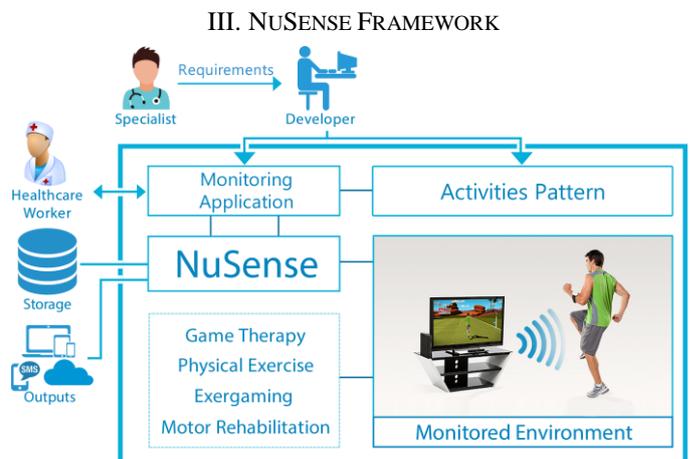


Figure 1. NuSense's Usage Scenario

We developed the NuSense, a sensor-based framework that allows building applications for the monitoring of environments focused on game therapies and motor rehabilitation. In Figure 1, we show the usage scenario of the NuSense Framework. To develop a solution to monitoring users, the specialist defines the activities that he want to be tracked during a therapy. According with the requirements, the developer will implement an integrated solution to NuSense, which will assist in the creation of the monitoring application. Finally, the healthcare worker will use the application to the

monitoring of users during activities like physical exercises, exergames, motor rehabilitation, or game therapies.

The framework should help to generate outputs to other devices, to storage the knowledge, and to make recommendations about the monitored activities. The application is created according with the sensors, which can be the KS, Sony EyeToy, mobile devices, webcams, body sensors and others. The framework provides a sensor interface allowing this flexibility. For every sensor, there is an associated plugin, which will perform the bridge between sensor and application. Initially we developed a plugin for the KS, and we are currently working on a plugin for Android devices.

Following, we show the NuSense’s architecture, and explain the modules functionality, according to Figure 2.

A. Communication Module

This module will be responsible for enable the interaction between sensor and application. Every sensor will need a predetermined plugin. The communication protocols used in the interaction between application and sensors are defined here. The collected data should be organized in a single format and sent to the Output Module.

B. Output Module

This module receives the encapsulated data from the Communication Module, and makes the data conversion according with the respective output, which could be image, text, bluetooth connection, cloud storage or another sensor.

C. Notification Module

While the activities are monitored and processed, if any anomaly is found, this module will create a register log. If the anomaly is related to the activity analysis, a recommendation should be sent to the Output Module, hence we can create an entry to another sensor or device such as smartphone. As example, if there are patients doing motor skill rehabilitation exercises at home, the doctor is able to receive the progress status on their smartphone, so he can give assistance to critical patients.

D. Storage Module

This module will store all the useful knowledge about the monitored activity. By default, the information is stored through a text log. A storage interface is provided to the developer, so he can store specific data, or use a specific DBMS.

E. Interfaces

In order to add a different sensor, the developer needs to implement the interface *ISensor*, according to Figure 2. The methods that must be implemented to create an extension point of the NuSense are described on Table I

The *IActivity* is the key-factor of the application, which is an interface to implement the features that need to be monitored. The Table II show the methods that should be implemented.

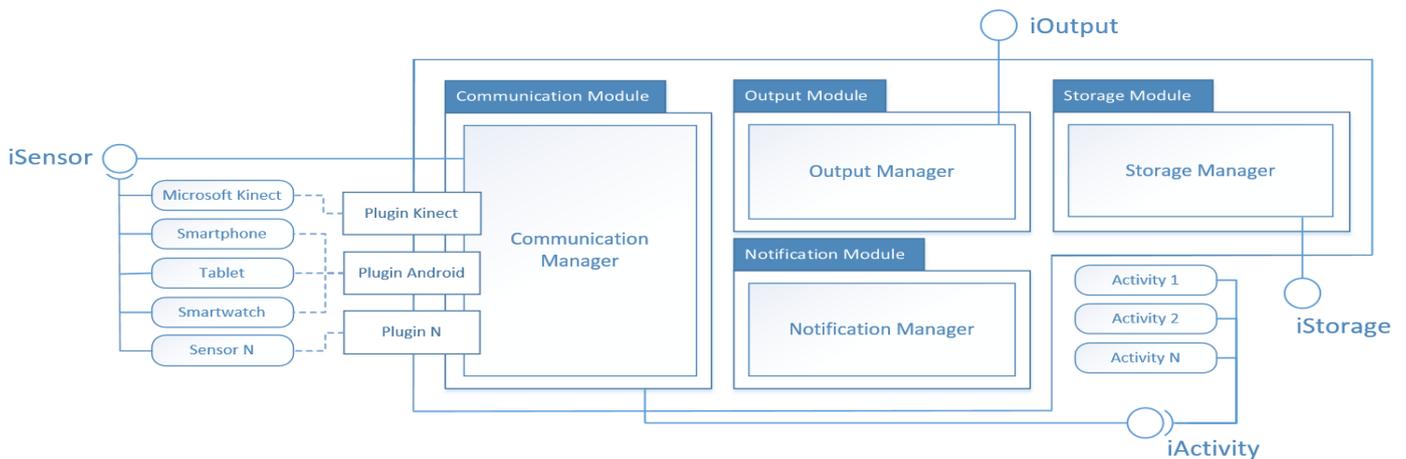


Figure 2. NuSense’s Architecture

TABLE I. DESCRIPTION OF THE ISENSOR INTERFACE

Method	Description
<i>initComponents</i>	Initialize the main entities of the component.
<i>finishTransmission</i>	Stops the data transmission.
<i>newDataAvailable</i>	Transmit the data available in the sensor.
<i>initActivity (IActivity)</i>	Initializes the activity analysis. <i>IActivity: Interface that defines the user behavior to be analyzed when he is doing an activity.</i>
<i>finishActivity</i>	Stops the activity analysis.
<i>getActivity</i>	Gets the activity properties.

TABLE II. DESCRIPTION OF THE IACTIVITY INTERFACE

Method	Description
<i>initComponents</i>	Initializes the main entities of the component.
<i>checkParameters (Object)</i>	Analyzes the user’s parameters and compare this with the activity pattern. <i>Object: Data provided by the sensor (e.g. user body coordinates)</i>
<i>sendAlertMessage</i>	Sends an alert to the notification module.

TABLE III. DESCRIPTION OF THE IOUTPUT INTERFACE

Method	Description
<i>getData(Object, type)</i>	Get the single format data from Communication Module. <i>Object: Encapsulated data.</i> <i>type: Object Enum.</i>
<i>dataConverter</i>	Converts data to a local output.
<i>dataConnection</i>	Sends data to a remote output.

To generate the data output, the interface IOutput was defined, which the Table III shows the required methods. Finally, for the storage of the analyzed information activity, the IStorage interface was provided, and the methods for their implementation can be found in Table IV.

TABLE IV. DESCRIPTION OF THE ISTOREAGE INTERFACE

Method	Description
<i>writeData (IActivity)</i>	Saves the parameters according with the application. <i>IActivity: Interface that defines the user behavior to be analyzed when he is doing an activity.</i>
<i>readData (id)</i>	Performs data reading. <i>Id: record identifier.</i>

IV. CASE STUDY: FLEXIBILITY ANALYSIS

For this case study we aim to create an application to benefit people with upper extremity dysfunction. The upper extremity dysfunction is common in people who have Brain Injury, Cerebral Palsy, Diabetes and Parkinson’s disease. This problem can affect the educational outcomes, the participations in activities of daily living, vocational options, etc. [25].

The specialists commonly use the flexibility test of body members to determine the presence of changes in the mobility of limbs, predict the natural history of the disease, evaluate changes in the range of motion (the range through which a joint can be moved) and check the treatment and disease progressions [26]. The purpose of evaluation is to determine the member's angular displacement. Currently this rate is measured by devices like the Fleximeter and Goniometer, but it is required that a qualified professional perform the measurement to ensure the accuracy of the test.

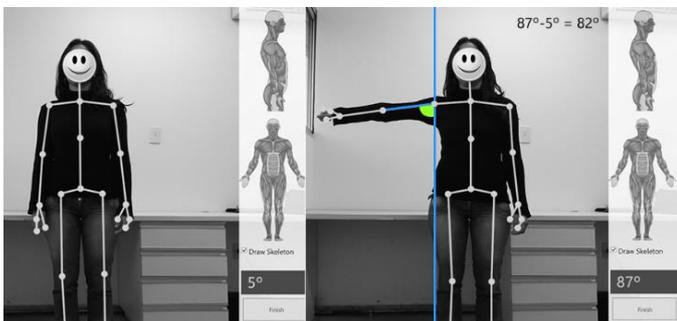


Figure 3. Shoulder’s Flexibility Application

The NuSense makes easier the process of building such application, because the developer can focus on the application

purpose. The benefits of the NuSense are related to the communication between application and KS. Using the KS, we do not care about the data gathering and the encapsulating of the information. Since the data are encapsulated, they are sent to the Output Module to be processed in the application with a *System.Drawing.Image* property from Microsoft .NET. In this case study, we developed the shoulder flexibility analysis, with the adduction and abduction of the shoulder, as illustrated in Figure 3.

To build this application, the framework was enhanced with a KS plugin, so the framework supports the KS 2.0, which use the Microsoft Kinect SDK 2.0.

The implementation of the adduction/abduction analysis aims to discover the angle below the shoulder, and uses the X and Y coordinates of the elbow and shoulder as described in the Equation I. The Listing 1 show the C# code of this analysis. After that, the angle is compared with the Leighton flexibility measure [27], and then the diagnosis is revealed.

$$\theta = \sin^{-1} \left(\frac{|Ex - Sx|}{\sqrt{((Ex - Sx)^2 + (Sy - Ey)^2)}} \right) \quad (I)$$

Ex: elbow coordinate X; **Ey:** elbow coordinate Y;
Sx: shoulder coordinate X; **Sy:** shoulder coordinate Y.

In the game context, the NuSense allows, during a gameplay, that the features defined by the specialist be mapped into the system. When the player performs an activity mapped at the application, an alert should be sent to the output module, which can generate a recommendation, such as adjustments of the therapy.

Another output was developed to send the monitored data to a mobile device. It is an Android version of the flexibility analysis application, which uses the KS as data gatherer and a desktop as data processor. Furthermore, we are developing another application with this structure focused on the posture of cyclists and bike fit. To develop such applications, it was needed only the development of the iOutput and iActivity interfaces, once the KS is still being the data gatherer.

V. CONCLUSIONS

In this paper, we presented the NuSense, a framework for support the development of applications for the automatic monitoring of people in environments applied to game-based therapies, motor rehabilitation and physical exercises.

This framework can help the developers to build such applications without concern about non-trivial concepts of technologies and techniques, such as different sensor integration, the data transmission and communication of the sensor with the solution.

We have detailed the modules functionalities and the interfaces that can be used according to the activity which will be monitored and the sensor type. It is expected that, with this framework, healthcare professionals be able to define features to create solutions to monitoring and assess users during a game therapy, solving problems such as divided attention.

As we demonstrated on our case study, the developers will have less work to create applications by using this approach. They should only implement the activities to be monitored,

leaving the data gathering of the sensor and the generation of outputs with the framework. Also, we notice that the NuSense is scalable, because every time a sensor or an output is added into the framework, the developers will be even more focused on the application domain, which is the main contribution of this work.

We are currently developing an extension of the NuSense, which supports Android device sensors as data gatherer, so it will be an Android plugin. As future work, we intend to focus on the synchronism of multiple sensors and the treatment of conflicting information between different sensors.

ACKNOWLEDGMENTS

This work was supported by the Center for Strategic Technologies in Health (NUTES) at the State University of Paraíba.

REFERENCES

- [1] Gil-Gómez, J. A.; Lloréns, R.; Alcañiz, M.; Colomer, C. Effectiveness of a Wii balance board-based system (eBaViR) for balance rehabilitation: a pilot randomized clinical trial in patients with acquired brain injury. *Journal of NeuroEngineering and Rehabilitation*. pp. 8:30, 2011.
- [2] Pirovano, M.; Lanzi, P. L.; Mainetti, R., & Borghese, N. A. The design of a comprehensive game engine for rehabilitation. *In Games Innovation Conference (IGIC), 2013 IEEE International*, p. 209-215, 2013.
- [3] De Urturi, Z. S.; Zorrilla, A. M.; Zapirain, B. G. Serious Game based on first aid education for individuals with Autism Spectrum Disorder (ASD) using android mobile devices. *In: Computer Games (CGAMES), 2011 16th International Conference on. IEEE*, p. 223-227, 2011.
- [4] Jaume-i-Capó, A.; Martínez-Bueso, P.; Mova-Alcover, B. & Varona, J. Interactive rehabilitation system for improvement of balance therapies in people with cerebral palsy. *Neural Systems and Rehabilitation Engineering, IEEE Transactions on*, 22(2), p.419-427, 2014.
- [5] Gago, J. R.; Barreira, T. M.; Carrascosa, R. G., & Segovia, P. G. Nutritional serious-games platform. *In eChallenges IEEE*, p. 1-8, 2010.
- [6] Szczesna, A.; Grudzinski, J.; Grudzinski, T.; Mikuszewski, R., & Debowski, A. The psychology serious game prototype for preschool children. *In Serious Games and Applications for Health (SeGAH), 2011 IEEE 1st International Conference on. IEEE*, p. 1-4, 2011.
- [7] Wattanasoontorn, V.; Boada, I.; García, R., & Sbert, M. Serious games for health. *Entertainment Computing*, 4(4), p.231-247, 2013.
- [8] Costa, T. H.; Soares, N. M.; Reis, W. A.; Bublitz, F. M. A Systematic Review on the Usage of Games for Healthcare.. *In: International Conference on Consumer Electronics (ICCE-Berlin), 2015 IEEE 5th International Conference on Consumer Electronics Berlin*, pp. 480-484, 2015.
- [9] Navarro, P.; Johns, M. L.; Lu, T. H.; Martin, H.; Poduval, V.; Robinson, M.; ... & Christel, M. G. Webz of war: A cooperative exergame driven by the heart. *In Games Innovation Conference (IGIC), 2013 IEEE International*, pp. 187-190) 2013.
- [10] Basu, D.; Moretti, G.; Gupta, G. S., & Marsland, S. Wireless sensor network based smart home: Sensor selection, deployment and monitoring. *In Sensors Applications Symposium (SAS), IEEE*, pp. 49-54, 2013.
- [11] Miniño, A. M.; Murphv, S. L.; Xu, J.; Kochanek, K. D. Deaths: final data for 2008. *National Vital Statistics Reports*. v.59, n.10, 2011.
- [12] Brandão, A., Trevisan, D. G., Brandão, L., Moreira, B., Nascimento, G., Vasconcelos, C. N., ... & Mourão, P. Semiotic inspection of a game for children with down syndrome. *In: Games and Digital Entertainment (SBGAMES), 2010 Brazilian Symposium on. IEEE*, pp. 199-210, 2010.
- [13] Kayama, H., Okamoto, K., Nishiguchi, S., Yamada, M., Kuroda, T., & Aoyama, T. Effect of a Kinect-based exercise game on improving executive cognitive performance in community-dwelling elderly: case control study. *Journal of medical Internet research*, 16(2), 2014.
- [14] Kerwin, M., Nunes, Francisco; Silva, Paula Alexandra. Dance! Don't Fall-preventing falls and promoting exercise at home. *Studies in health technology and informatics*, v. 177, p. 254-259, 2011.
- [15] O'Donovan, C., Roche, E. F., & Hussey, J. The energy cost of playing active video games in children with obesity and children of a healthy weight. *Pediatric obesity*, 9(4), p. 310-317, 2014
- [16] Maddison, R., Mhurchu, C. N., Jull, A., Prapavessis, H., Foley, L. S., & Jiang, Y. Active video games: the mediating effect of aerobic fitness on body composition. *Int J Behav Nutr Phys Act*, 9(1), p. 54, 2012.
- [17] Zhang, M., Xu, M., Liu, Y., He, G., Han, L., Lv, P., & Li, Y. The framework and implementation of virtual network marathon. *In VR Innovation (ISVRI), 2011 IEEE International Symposium on*, pp. 161-167, 2011.
- [18] Alamri, A., Hassan, M. M., Hossain, M. A., Al-Ourishi, M., Aldukhavvil, Y., & Hossain, M. S. Evaluating the impact of a cloud-based serious game on obese people. *Computers in Human Behavior*, 30, p. 468-475, 2014.
- [19] Stach, C., & Schlindwein, L. F. M. Candy Castle—A prototype for pervasive health games. *In: Pervasive Computing and Communications Workshops (PERCOM Workshops), 2012 IEEE International Conference on*, pp. 501-503, 2012.
- [20] De Bortoli, A., & Gaggi, O. PlayWithEyes: A new way to test children eyes. *In Serious Games and Applications for Health (SeGAH), 2011 IEEE 1st International Conference on*. p. 1-4, 2011.
- [21] Konstantinidis, E. I., Antoniou, P. E., Bampanopoulos, G., & Bamidis, P. D. A lightweight framework for transparent cross platform communication of controller data in ambient assisted living environments. *Information Sciences*, 300, p. 124-139, 2015.
- [22] Apiletti, D., Baralis, E., Bruno, G., & Cerquitelli, T. Real-time analysis of physiological data to support medical applications. *Information Technology in Biomedicine, IEEE Transactions on*, 13(3), p. 313-321, 2009.
- [23] Ivengar, S., Bonda, F. T., Gravina, R., Guerrieri, A., Fortino, G., & Sangiovanni-Vincentelli, A. A framework for creating healthcare monitoring applications using wireless body sensor networks. *Proceedings of the ICST 3rd international conference on Body area networks*. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), p. 8, 2008.
- [24] Pirovano, M., Mainetti, R., Baud-Bovy, G., Lanzi, P. L., & Borghese, N. A. IGER-intelligent game engine for rehabilitation. *IEEE Trans. Comput. Intell. AI Games*, p. 1, 2014.
- [25] Boyd, R. N., Morris, M. E., & Graham, H. K. Management of upper limb dysfunction in children with cerebral palsy: a systematic review. *European Journal of Neurology*, 8(s5), p. 150-166, 2001.
- [26] Bower, E., Ashburn, A., & Stokes, M. Princípios de conduta fisioterapêutica e medidas de resultado final. *Neurologia para fisioterapeutas*. São Paulo: Premier, p.49-63, 2000.
- [27] Leighton, J.R. Manual of instruction for Leighton Flexometer. New York: Human Kinetics, 1987.

Modeling and analyzing cost-aware fault tolerant strategy for cloud application

Liqiong Chen^{1,2}, Guisheng Fan³, Yunxiang Liu¹

¹Department of Computer Science and Information Engineering
Shanghai Institute of Technology, Shanghai 200235, China

²Shanghai Key Laboratory of Computer Software Evaluating and Testing, Shanghai 201112, China

³ Department of Computer Science and Engineering
East China University of Science and Technology, Shanghai 200237, China
Correspondence should be addressed to Guisheng Fan; gsfan@ecust.edu.cn

¹ **Abstract**—In this paper, we propose a method to model and analyze cost-aware fault tolerant strategy for cloud computing. First, Petri nets are used to describe the structure of cloud computing, including component, cloud service and cloud application, thus forming the fault tolerant model of cloud computing. Second, a dynamic fault tolerant strategy is proposed, which can dynamically make fault tolerant strategy with the lowest cost based on the current state and failed component. Third, we present operational semantics and related theories of Petri nets for establishing the correctness of our proposed method. We have also performed a series of simulations to evaluate our proposed approach. Results show that it can help reveal the structural and behavioral characteristics of cloud computing, and reduce the fault tolerant cost.

Index Terms—Cloud computing, fault tolerance, cost, Petri net, reliability

I. INTRODUCTION

Cloud computing sets a new paradigm for infrastructure management by offering unprecedented possibilities to deploy software in distributed environments[1]. While fault tolerant processing techniques are mainly used for the development of reliable distributed systems [2]. However, cloud computing may include a number of cloud applications. The fault tolerant strategy of cloud application may affect each other, and different strategies may have different costs. It is a challenging task to dynamically make the fault tolerance strategy with the lowest cost for the failed cloud components. In addition, most of the previous methods on fault tolerant process for cloud application didn't consider the user's QoS constraints (such as time, cost, etc.). If the cost of cloud application is high, then the users will be unwilling to use cloud service, which in turn would cause the loss of users' interest in the cloud service.

This paper investigates how to model and analyze cost aware fault tolerant strategy for cloud computing. Below summarizes our main contributions: First, we provide a flexible way for designers to specify their requirements, Petri nets are used to model different components of cloud computing. Second, we propose a method to dynamically make fault tolerant strategy, which can get the fault tolerant strategy with the lowest cost based on the current state and failed component. The process that many cloud applications compete for cloud service is converted into the optimization of fault tolerant strategy by considering the recovery cost. Third, the operational semantics and related theories of Petri nets help establish the effectiveness of our proposed method.

The remainder of this paper is organized as follows. Section II describes how we model the different components of cloud application. Next, Section III proposes the fault tolerant strategy and analysis technique, and then evaluate the proposed method via a series of simulations (Section IV). Section V surveys related work, and Section VI concludes.

II. MODELING CLOUD APPLICATION

A. Requirements of cloud computing

Because cloud computing may include several cloud applications. As the function of cloud application is composed by a number of independent sub-functions (component) according to a certain composition rules [3], each component has several cloud services which can realize its function.

Definition 1: The fault tolerant requirement of cloud application is 6-tuple: $\Xi = (C, WS, A, RC, RW, RL)$: (1) C is the finite set of component in cloud application. (2) $WS = (WS^c, WS^e)$ is the finite set of service, WS^c, WS^e are the set of matching service and replacement service. (3) A is the finite set of cloud application. (4) $RC : C \rightarrow (0, 1)$ is the weight of component. (5) $RW = (CC, CE)$ is the attribute function of service, CC is the reliability of service. $CE(WS_i^e) = (ae_i, se_i, at_i, st_i)$ is the cost, starting cost, running time and start-up time of replacement service. (6) $RL = (RLr, RLe, RLc)$, RLr is the relation function between the components. RLe, RLc are the set of replacement service and matching service of component.

The fault tolerant strategy of cloud application is composed by a series of replacement services. $SP = (SP_1, \dots, SP_f)$ is the fault tolerant strategy of cloud applications a_1, \dots, a_f . $SP_i = \{WS_k, \dots, WS_g\}$ describes the replacement service of all components in a_i .

B. Syntax and semantics

Petri net (PN) is a formal language for describing the distributed system because its semantics is formally defined[4].

Definition 2: A 6-tuple $\Sigma = (PN, IO, tr, D, A_F, M_0)$ is called a Fault tolerant Model(FTM): (1) $PN = (P, T, F, W)$ is a basic Petri net. P, T, F, W are the finite set of place, transition, arc and weight. (2) IO is a special type of place, which is the interface of Σ and denoted by dotted circle. (3) tr is the attribute function of transition, $tr(t_i) = (\lambda_i, pi_i, r_i, ct_i)$ is the firing probability(deterministic), cost, priority and running time of transition, the default value is (1, 0, 0, 0) The smaller the value of r_i , the higher the priority of transition, the priority of instantaneous transition is higher than time transition. (4)

¹DOI reference number: 10.18293/SEKE2016-247

$D = \{d_{i,j}^c, d_k^w, \varphi\}$ is the individuality of component, the matching service and data packet φ . (5) A_F is the formula set or individuality on the arc. (6) M_0 is the initial marking of FTM .

The input/output arc of transition t_i and the free variable in $A_T(t_i)$ are denoted by $FV(t_i)=\{x_1, x_2, \dots, x_n\}$. $S=(M, EC, ET)$ is the state of model, EC and ET are the cost and time when the system reaches S , the initial state $S_0=(M_0, EC_0, ET_0)$, $EC_0=ET_0=0$. $\forall t \in T$, if $FV(t_i)=\{x_1, x_2, \dots, x_n\}$, the token $\{d_1, d_2, \dots, d_n\}$ meets $d_i \in \{M(p) \mid p \in \bullet t \cup t^*\}$ and d_i corresponds to the variable x_i , the instance of t is obtained by replacing d_1, d_2, \dots, d_n with x_1, x_2, \dots, x_n , which is called a replacement of t , denoted by $t < d_1, d_2, \dots, d_n >$, it can be denoted by $t < d_1, d_2, \dots, d_n >$. The replacement is mainly used to bind the token to the input/output arc of t and all free variables in $A_T(t)$. Let $A_T(t) < d_1, d_2, \dots, d_n >$ and $A_F(p, t) < d_1, d_2, \dots, d_n >$ be the values got by replacing $A_T(t)$ and $A_F(p, t)$ of input arc with d_1, d_2, \dots, d_n .

Definition 3: If $t < d_1, d_2, \dots, d_n >$ makes $A_T(t) < d_1, d_2, \dots, d_n > \wedge A_F(t) < d_1, d_2, \dots, d_n > = \text{true}$, then $t < d_1, d_2, \dots, d_n >$ is called the feasible replacement of transition t under state S .

All the feasible replacements of transition t under S are denoted by set $VP(S, t)$. If $VP(S, t) \neq \emptyset$, then t is enable under S , denoted by $S[t >]$. Let $ET(S) = \{t \mid S[t >]\}$. For $t_i \in ET(S)$, if $\lambda_i \leq \min(\lambda_j)$, $t_j \in ET(S)$, then the firing of t_i under S is effective. All effective firing transitions under state S are denoted by $FT(S)$. The process that S reaches S' by firing a feasible replacement $t_i < d_1, d_2, \dots, d_n >$ of t_i is denoted by $S[t_i < d_1, d_2, \dots, d_n > S']$. If the relation between transitions is parallel, then the firing of any transition cannot affect the firing of another transition. The concurrent transitions under S are denoted by set $MT(S)$.

Definition 4: $H(S)=\{t < d_1, d_2, \dots, d_n > \mid t \in MT(S), t < d_1, d_2, \dots, d_n > \in VP(S, t) \wedge (ct_i \leq \min(ct_f), t_f \in MT(S')) \wedge \lambda_i = \infty\}$ is called the greatest firing set of S .

FTM can reach a new state S' by firing the greatest firing set $H(S)$ under the state S .

Definition 5: Let Ω be a FTM model, S is a state of Ω , the system can reach a new state S' by effectively firing $H(M)$, denoted by $S[H(S) > S']$, then S' is called a reachable state of S . S' is computed based on the following rules:

(1) $\forall t_i < d_1, d_2, \dots, d_n > \in H(S), \forall p_j \in \bullet t_i \cup t_i^* : M'(p_j) = M(p_j) - A_F(p_j, t_i) < d_1, d_2, \dots, d_n > + A_F(t_i, p_j) < d_1, d_2, \dots, d_n >$.

(2) Computing EC' : $EC' = EC + \sum_{t_i \in H(S)} \pi_i$.

(3) Computing ET' : $ET' = ET + \max\{ct_i\}, t_f \in H(S)$.

All the possibly reachable states of S are denoted by $R(S)$. FTM will start from the initial state S_0 and generate the new state by effectively firing the enabled transitions. $\delta(S_i, S_j)$ is the firing sequence from S_i to S_j .

C. Modeling basic elements

Modeling component. The FTM model of component $C_{i,j}$ is shown in Fig.1, where $D = \{d_{i,j}^c, d_k^w, \varphi\}$ is the individuality of component, which represents the component $C_{i,j}$, the matching service WS_k and data packet φ . The initial resource distribution $M_0(p_{ws,i,j}) = d_k^w$. A_F is the formula set or individuality on the arc, such as $A_F(p_{ws,i,j}, t_{s,i,j}) = x$.

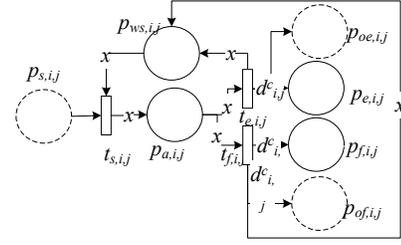


Fig. 1. Modeling component

Modeling replacement service. The FTM model of WS_i is shown in Fig.2, tr is the attribute function of transition. so the firing time of $t_{e,i}$ is equal to the execution time of service. The initial distribution of resources is $M_0(p_{I,i}) = d_i^w$.

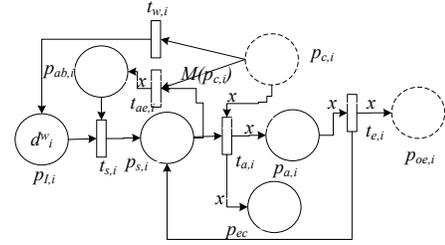


Fig. 2. Modeling replacement service

Modeling cloud application. The steps for constructing cloud application are as follows. 1) Matching the resources for each component based on the requirement, then construct the model of all components. 2) Introducing $t_{s,i}$ and $p_{s,i}$ to describe the beginning operation and position of the application, then initialize the system based on the characteristics of cloud application. At the same time, we introduce $t_{e,i}$ and $p_{e,i}$ to describe the termination operation and position of the system. 3) Composing the model of each element based on the basic relationships between each element. 4) Introducing $p_{af,i}$ to store all failed component in a_i . 5) Setting $M_0(p_{s,i}) = \varphi$.

Modeling fault tolerant processing. $t_{af,i}$ is used to upload all failed components in cloud application a_i to the place p_{fc} . p_{ench} is used to store the fault tolerant strategy of each cloud application. If any component fails ($|p_{fc}| \neq 0$), then fire t_{ench} to allocate the component to the replacement service according to the fault tolerant strategy. $t_{oe,i}$ is used to transfer the results of replacement service WS_i to the cloud application.

Modeling cloud computing $\Omega(sp_1^i, sp_2^k, \dots, sp_l^f)$. The steps for constructing cloud computing are as follows. 1) Constructing the fault tolerant model of component, replacement service and cloud application. 2) Modeling the adopted fault tolerant strategy for cloud computing, sp_j^i represents that a_j takes the i -th fault tolerant strategy, $sp_j^i = \{WS_{s_k}, \dots, WS_{s_m}\}$. 3) Merging the same place and transition, then set the initial marking.

III. FAULT TOLERANT STRATEGY

A. Fault tolerant cost

Let the current fault tolerant strategy be sp_d . And the configured replacement service of $C_{i,j}$ under sp_d is WS_{s_f} , which is denoted by $exC(C_{i,j}, sp_d) = WS_{s_f}$. The principle of fault tolerant strategy is to select the replacement service for failed component, so $WS_{s_f} \in RW(C_{i,j})$ and WS_{s_f} is unique.

$exW(W S_f, sp_d) = \{C_{i,j} | C_{i,j} \in C \wedge exC(C_{i,j}, sp_d) = W S_f\}$ is called the finished component set of $W S_f$ under sp_d .

The recovery time of component $C_{i,j}$:

$$Alt(C_{i,j}, sp_d) = \begin{cases} at_f + st_f, & exW(W S_f, sp_d) = C_{i,j}; \\ \frac{st_f + at_f \times |exW(W S_f, sp_d)|}{|exW(W S_f, sp_d)|}, & else \end{cases}$$

In the same way, the recovery cost of component $C_{i,j}$:

$$Ale(C_{i,j}, sp_d) = \begin{cases} se_f + ae_f, & exW(W S_f, sp_d) = C_{i,j}; \\ \frac{se_f + ae_f \times |exW(W S_f, sp_d)|}{|exW(W S_f, sp_d)|}, & else \end{cases}$$

The above formula computes the recovery cost and time of component from the view of the reliability of matching service. The recovery of component must consider the time and cost. We will give the recovery cost of component in the following:

The fault tolerant cost of $C_{i,j}$ under sp_d is: $Al(C_{i,j}, sp_d) = x_1 \times \frac{max_Alt(C_{i,j}) - Alt(C_{i,j})}{max_Alt(C_{i,j}) - min_Alt(C_{i,j})} + X_2 \times \frac{max_Ale(C_{i,j}) - Ale(C_{i,j})}{max_Ale(C_{i,j}) - min_Ale(C_{i,j})}$, where $x_1 + x_2 = 1$

The fault tolerant cost of a_i under sp_d is equal to the fault tolerant cost of failed components:

$$Al(a_i, sp_d) = \prod_{C_{i,j} \in C_i} RW(C_{i,j}) \times Al(C_{i,j}, sp_d)$$

The fault tolerant cost of cloud application under the strategy sp_d is: $Al(sp_d) = \prod_{a_i} Al(a_i, sp_d)$

B. Dynamic fault tolerant strategy

Let the initial value of current fault tolerant strategy sp_d be $\{\emptyset, \emptyset, \dots, \emptyset\}$. We will make dynamic fault tolerant strategy based on the following steps when component $C_{i,j}$ fails.

(1) $\forall W S_f \in RW(C_{i,j})$, let $sp_{temp} = (sp_1^t, sp_2^t, \dots, sp_{i-1}^t, sp_{i,f}^t, \dots, sp_k^t)$, that is, the system will select service $W S_f$ for $C_{i,j}$ to recovery under sp_i^t , then compute $Al(C_{i,j}, sp_{i,f}^t)$.

(2) The system will select $W S_f$ with the minimum value of $Al(C_{i,j}, sp_{i,f}^t)$ as the replacement service of $C_{i,j}$: $sp_d = (sp_1^d, sp_2^d, \dots, sp_{i-1}^d, sp_i^d, \dots, sp_k^d)$, where $sp_i^d = sp_i^d \cup \{(C_{i,j}, W S_f)\}$, and sp_d will be viewed as the current strategy.

According to the above steps, we can ensure that cloud computing can dynamically consider the fault tolerant cost. We can weave the dynamic fault tolerant strategy into the transitions of fault tolerant model.

Definition 6: Let Ω be a fault tolerant model, S is a state of Ω , $H(S) = \{t < d_1, d_2, \dots, d_n \mid t \in MT(S), t < d_1, d_2, \dots, d_n \in VP(S, t)\}$ be the greatest concurrent set of S , $M(p_{ench}) = d_d^s, d_d^s$ is the token of current strategy. We can further set $H(S)$ according to the dynamic fault tolerant strategy: If $\forall t_{f,i,j} \in FT(S)$, $Al(C_{i,j}, sp_{i,f}^t) = \min\{Al(C_{i,j}, sp_{i,k}^t)\}$, $d_k^w, d_f^w \in \#2(d_{i,j}^c)$, then $F(t_{ench}, p_{c,f}) < x \leftarrow d_{i,j}^c >$

We will analyze the correctness of dynamic fault tolerant strategy based on the internal mechanism of cloud computing.

Theorem 1: Let Ω be the fault tolerant model of cloud application, $R(\Omega)$ be the reachable state set which is obtained by using fault tolerant strategy. $\forall \in R(\Omega)$, $EW(S)$ is the firing set of cloud service when the system reaches S , $\forall W S_i \in EW(S)$, $W S_i$ is the replacement service of $C_{f,k}$ then: $\forall W S_j \in WS - EW(S)$, if $C_{f,k} \in RC(W S_i) \cap RC(W S_j)$, then $Al(C_{f,k}, sp_{f,i}^t) \leq Al(C_{f,k}, sp_{f,j}^t)$.

Proof by contradiction: $\exists W S_i \in EW(S)$, $\exists W S_j \in WS - EW(S)$, $C_{f,k} \in RC(W S_i) \cap RC(W S_j)$, then $Al(C_{f,k}, sp_{f,i}^t) > Al(C_{f,k}, sp_{f,j}^t)$. Because $W S_i \in EW(S)$,

TABLE I
RESOURCE CONFIGURATION AND ATTRIBUTES

C	Actual meaning	CC	RA
$C_{1,1}, C_{2,1}, C_{3,1}$	Requirement analysis	99%	$W S_1, W S_2, W S_3$
$C_{1,2}, C_{4,1}$	Loading optimization	98%	$W S_4, W S_5, W S_6$
$C_{1,3}, C_{2,2}, C_{4,2}$	Route optimization	99%	$W S_7, W S_8, W S_9$
$C_{1,4}, C_{2,3}, C_{4,3}$	Network optimization	98%	$W S_{10}, W S_{11}, W S_{12}$
$C_{1,5}, C_{3,2}, C_{4,4}$	Payment completion	98%	$W S_{13}, W S_{14}, W S_{15}$
$C_{1,6}, C_{3,3}$	Transportation monitoring	97%	$W S_{16}, W S_{17}, W S_{18}$
$C_{1,7}, C_{3,4}$	Transportation querying	98%	$W S_{19}, W S_{20}$
$C_{3,5}$	Transportation route navigation	99%	$W S_{21}$
$C_{1,8}, C_{2,4}, C_{3,6}$	Late, and other service	97%	$W S_{22}, W S_{23}, W S_{24}$

TABLE II
ATTRIBUTE OF REPLACEMENT SERVICE

WS	ae	se	at	st	WS	ae	se	at	st
$W S_1$	2	3	2	1	$W S_{13}$	2	2	2	2
$W S_2$	3	2	2	2	$W S_{14}$	3	3	1	3
$W S_3$	2	2	2	3	$W S_{15}$	3	4	2	3
$W S_4$	4	2	2	1	$W S_{16}$	4	3	3	2
$W S_5$	5	4	1	2	$W S_{17}$	3	2	1	3
$W S_6$	2	2	2	2	$W S_{18}$	4	3	2	3
$W S_7$	2	3	2	3	$W S_{19}$	2	4	1	2
$W S_8$	3	2	3	3	$W S_{20}$	2	2	2	2
$W S_9$	3	3	1	1	$W S_{21}$	4	3	3	3
$W S_{10}$	4	2	2	2	$W S_{22}$	3	4	3	4
$W S_{11}$	2	3	2	3	$W S_{23}$	4	4	2	2
$W S_{12}$	2	3	3	3	$W S_{24}$	2	3	2	3

S_1 is in one of the firing sequence from S_0 to S , which makes $t_{ench} \in FT(S_1)$. Because $C_{f,k} \in RC(W S_i) \cap RC(W S_j)$, therefore $d_{f,k}^c \in S(p_{fc})$. According to the definition of feasible replacement, we can get $t_{ench} < x \leftarrow d_{f,k}^c >$ and $t_{ench} < y \leftarrow d_{f,k}^c >$ are two feasible replacements of t_{ench} . Because $Al(C_{f,k}, sp_{f,i}^t) > Al(C_{f,k}, sp_{f,j}^t)$, according to the Definition 3, we can get $H(S_1) = H(S_1) \wedge t_{ench} < x \leftarrow d_{f,k}^c >$, therefore, $\exists W S_j \in EW(S)$, which is contradicted with $W S_j \in WS - EW(S)$, the assumption does not establish.

Theorem 1 illustrates that the proposed method can ensure that fault tolerant cost of dynamically selected component is the locally optimal value, the function is to select the schema with the lowest cost in the execution process.

IV. EXAMPLE

In this paper, we use a simplified logistics cloud as an example. Four logistics clouds are operating at the same time, because each application has the different purpose, its execution processes are different too: $a_1 : C_{1,1} > C_{1,2} > (C_{1,3} \parallel C_{1,4}) > C_{1,5} > (C_{1,6} + C_{1,7}) > C_{1,8}$; $a_2 : C_{2,1} > (C_{2,2} \parallel C_{2,3}) > C_{2,4}$; $a_3 : C_{3,1} > C_{3,2} > (C_{3,3} + C_{3,4} + C_{3,5}) > C_{3,6}$; $a_4 : C_{4,1} > (C_{4,2} \parallel C_{4,3}) > C_{4,4}$. The attributes of component are shown in Table I. The weight of components in application is $\{0.1, 0.1, 0.2, 0.1, 0.1, 0.1, 0.2, 0.1\}$, $\{0.3, 0.3, 0.2, 0.2\}$, $\{0.2, 0.1, 0.2, 0.2, 0.1, 0.2\}$, $\{0.2, 0.3, 0.4, 0.1\}$. The system has 24 replacement services, their attributes are shown in Table II.

We can construct the fault tolerant model of replacement service, cloud application and cloud computing in the same way. We can verify the related properties of model by using the related tools of Petri nets, which includes the correctness of execution process and fault tolerant strategy, the selection of replacement service. Based on the state space of fault tolerant model, we can get that the state space is limited, and the applications can reliably operate when the component fails. We can randomly generate 15 fault tolerant strategies. i represents the service $W S_i$. First, we can compute the cost of $C_{1,1}, C_{2,4}, C_{3,6}, C_{3,5}$ under the different strategies, which is shown in Fig.3(a). We can get that the cost of $C_{2,4}, C_{3,6}$ under the same strategy is different even if the actual meaning and the replacement service of component are same. The cost

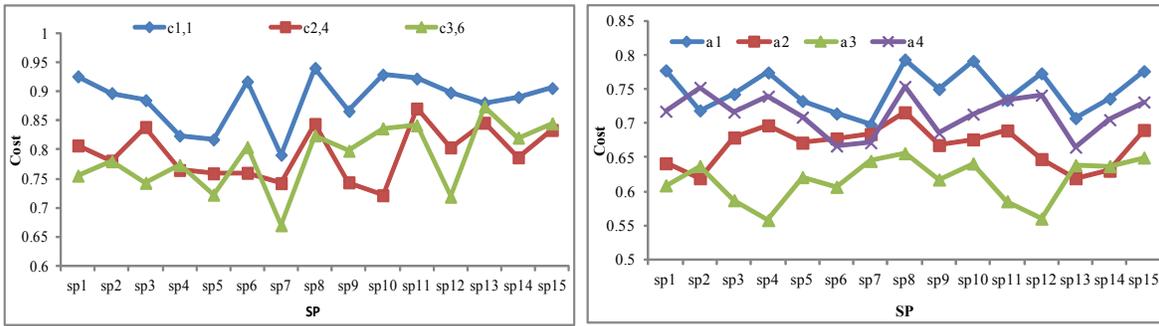


Fig. 3. Fault tolerant cost of application under different strategies

of all applications under the different strategy is shown in Fig.3(b). We can get that: (1) For the same application, the cost is different under the different strategies due to the cost of component is different. And all applications can get the lower cost under the strategy SP_8 . (2) Not all applications can get the lowest cost under SP_8 .

V. RELATED WORKS

Fault tolerance is an important means to improve the software reliability. Reference [6] proposes a large-scale fault injection system that can execute numerous model-based fault-injection simulations in a reasonable time using a cloud computing environment. A scalable hybrid Cloud infrastructure as well as resource provisioning policies to assure QoS targets of the users is presented in [7]. In order to increase the fault tolerance of cloud computing, a number of researcher and institutions begin to focus on cloud computing fault-tolerant framework [8], and to further explore the Byzantine fault -oriented cloud computing architecture [9]. However, several aspects differentiate our approach from the above approaches. First, the modeling and analysis process is easier to use because of the high abstraction level offered by using formal method, which help in strengthening the flexibility of composition process. Second, we propose the dynamic fault tolerant strategy, which can guarantee that cloud computing can select the optimal cloud service to realize the function of failed component, thus reducing the cost.

Many research efforts for cloud computing have adopted formal methods techniques to leverage its mathematically precise foundation for providing theoretically sound and correct formalisms. Bruneo, D. et al. propose a technique to model and evaluate the VMM aging process and to investigate the optimal rejuvenation policy that maximizes the VMM availability under variable workload conditions [10]. Ghosha et al. [11] develop a scalable stochastic analytic model for performance quantification of Infrastructure-as-a-Service (IaaS) Cloud. Reference [12] presents a framework called E-mc2 for modelling the energy consumption in cloud computing system. In contrast, we have proposed an approach to constructing the reliable service composition, which provides means to observe behaviors of basic component, and to describe their interrelationship [13]. Most of the aforementioned formalisms cover basic and structured activities of cloud application, but they are unable to ensure that the constructed model can meet the users' requirements, such as cost and reliability.

VI. CONCLUSION

In this paper, Petri nets are used to describe different components of cloud computing. The reliability and cost are took into account in the modeling process. Then, we propose a method to dynamically make fault tolerant strategy, which can get the fault tolerant strategy with the lowest cost based on the current state and failed component. Third, we present the operational semantics and related theories of Petri nets to help prove the effectiveness of proposed method,. Finally, we also conduct experiments to evaluate the proposed method.

ACKNOWLEDGMENT

The work is partially supported by the NSF of China under grants No. 61173048 and 61300041. Research Fund for the Doctoral Program of Higher Education of China under Grants No. 20130074110015.

REFERENCES

- [1] S. Marstona, Z. Lia, S. Bandyopadhyaya, et al. Cloud computing-the business perspective. *Decision Support Systems*. 2011, 51(1): 176-189.
- [2] B. Yang, F. Tan, Y. S. Dai. Performance evaluation of cloud service considering fault recovery. *The Journal of Supercomputing*. 2013, 65(1):426-444.
- [3] Wang, X. Y, Du, Z. H, Chen, Y. N. An adaptive model-free resource and power management approach for multi-tier cloud environments. *Journal of Systems and Software*. 2012, 85(5): 1135-1146.
- [4] M. TADAO. Petri nets: properties, analysis and application. *Proceedings of the IEEE*. 1989, 77(4):540-581.
- [5] R. N. Calheiros, R. Ranjan, A. Beloglazov, C. A. F. De Rose, R. Buyya. CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms. *Software: Practice and Experience*. 2011, 41(1):23-50.
- [6] Y. Nakata, Y. Ito, Y. Takeuchi, et al. Model-based fault injection for large-scale failure effect analysis with 600-node cloud computers. http://cs28.cs.kobe-u.ac.jp/assets/files/pdf/1303_nakata_RIIF.pdf.
- [7] B.Javadi, J. H. Abawajy, R. Buyya. Failure-aware resource provisioning for hybrid Cloud infrastructure. *Journal of Parallel and Distributed Computing*. 2012, 72(10): 1318-1331.
- [8] G. Belalem, S. Limam. Fault tolerant architecture to cloud computing using adaptive checkpoint. *International Journal of Cloud Applications and Computing*. 2011, 1(4): 60-69.
- [9] Y. Zhang, Z. Zheng, M. R. Lyu. BFTCloud: A byzantine fault tolerance framework for voluntary-resource cloud computing. *Processing of the 2011 IEEE International Conference on Cloud Computing*. IEEE Computer Society, Washington, DC, USA, 2011: 444-451.
- [10] D. Bruneo, S. Distefano, F. Longo, A. Puliafito, M. Scarpa. Workload-based software rejuvenation in cloud systems. *IEEE Transactions on Computers*. 2013, 62(6):1072-1085.
- [11] R. Ghosha, F. Longob, V. K. Naik, K. S. Trivedi. Modeling and performance analysis of large scale IaaS Clouds. *Future Generation Computer Systems*. 2013, 29(5):1216-1234.
- [12] G. C. Gabriel, N. Alberto, L. Pablo, et al. E-mc2: A formal framework for energy modelling in cloud computing. *Simulation Modelling Practice and Theory*. 2013, 39(2013): 56-75.
- [13] G. Fan, H. Yu, L.Chen, D.Liu. Petri net based techniques for constructing reliable service composition. *Journal of Systems and Software*. 2013, 86(4): 1089-1106.

From Design to Code: An Educational Approach

Candice Eckert*, Brian Cham*, Jing Sun[†] and Gillian Dobbie[†]

*Department of Electrical and Computer Engineering

[†]Department of Computer Science

The University of Auckland, New Zealand

Emails: *{ceck002, bcha899}@aucklanduni.ac.nz, [†]{jing, gill}@cs.auckland.ac.nz

Abstract—Model Driven Engineering (MDE), despite having many advantages, is often overlooked by programmers due to lack of proper understanding and training in the matter. This paper investigates the advantages and disadvantages of MDE and looks at research results showing the adoption rates of design models. In light of the findings, an educational tool, namely Lorini, was developed to provide automated code generation from the design models. The implemented tool consists in a plug-in for the Astah framework aimed at teaching Java programming to students through UML diagrams. It features instantaneous code generation from three types of UML diagrams, code-diagram matching, a feedback panel for error displays and on-the-fly compilation and execution of the resulting program. Evaluation of the tool indicated it to be successful with unique educational features and intuitive to use.

I. INTRODUCTION

The development of a software product includes many steps such as requirement analysis, design, coding and testing. The design phase usually involves formal design models. A more efficient alternative to manual coding is automated code generation from design models. This approach is known as Model Driven Engineering (MDE). The main advantages of MDE are its efficiency in terms of development time[1] and the reduction of human errors. This last property plays an important role in real-time embedded systems, where the slightest mistake could lead to lethal consequences[2]. However, the structure of auto-generated code is more complex than manual code, making it harder to understand and re-use[3]. It is also less efficient in terms of number of calls per executable statement, number of executable statements, and time taken by stack allocation/deallocation[1].

The main issue regarding MDE remains the low adoption rate. Only 11% of programmers often use formal design models, others using it scarcely, for communication purposes only, or not at all[4]. This is mostly due to a lack of understanding and training in the matter. An experiment conducted on a software design class showed that most students did not use MDE before the course, and 70% of them found MDE to be useful in order to understand software design[5]. By combining these research results, the emerging factor seems to be the lack of educational tools for programmers to learn formal design modelling. This has the unfortunate effect

of reducing the adoption rate of MDE, even though many improvements could be introduced by using design models.

Giving a new direction to the research, it was found that teaching modelling before programming is feasible and even beneficial, considering that modelling is applicable to different disciplines and can favour a more structured learning[6]. It is suspected that complex object oriented concepts could be more easily grasped by students if they were introduced through design models. As a consequence of these findings, we decided to develop an educational tool for students to learn object oriented programming starting from formal design models, specifically learning Java from UML. An overview block diagram of the implemented tool is shown in Figure 1.

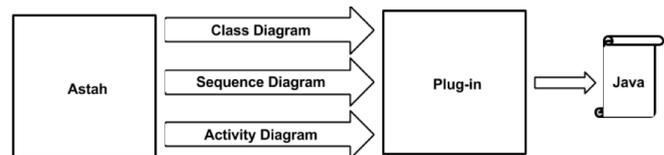


Fig. 1. Overall Approach to UML to Code Generation

The tool adopts the approach described by Rivera-Lopez et al.[7] in teaching the foundations of object oriented programming using UML. We make use of three types of UML diagrams, namely class, sequence and activity diagrams, and the following generation steps:

- Design the class diagram showing the relationships among the classes, and generate the Java class skeleton code from the class diagram.
- Design the sequence diagram describing the method calls from one class to another, and generate the main class (the main program) using the sequence diagram.
- Design the activity diagram corresponding to the code of each method, and generate the methods of the Java class from the activity diagrams.

It allows the automatic generation of executable Java program from UML design models, and visually highlights the relationships between the design model and the code. It is important to note that the implemented tool is not intended to be used in place of a programming course but rather as a supplementary learning aid aiming to facilitate the students' understanding of the subject. There are many different systems and plug-ins that perform code generation from design models.

After a brief analysis of thirty-five of them, a number of free, open-source tools were extracted and investigated in more depth. Some tools were found to be too complicated to use or difficult to install due to lack of documentations. With careful consideration, the Astah framework was elected as the best option. Its interface is user-friendly and easy to understand for intuitive users. A clear set of APIs is available, making it easily extendible. Moreover a free licence is dispensed to students, thus facilitating both the design and evaluation processes. Astah is available on Windows, Linux and Mac OS X, and the plug-ins are easy to install. The tools used were Eclipse IDE for code development and a Github repository for version control, helping with collaboration and synchronization. The outcome of the project was an interactive plug-in for the Astah modelling framework.

The rest of the paper was structured as follows. Section II discusses the findings of related research in the field, including surveys and existing solutions. In section III, we outline the objectives and technical decisions behind the plug-in to explain the rationale of the project. Section IV describes the implementation details of the plug-in to explain how it was developed. In section V, we list the criteria and conduct evaluations to assess the success of the plug-in. Finally, section VI concludes the paper and outlines the future work.

II. LITERATURE REVIEW

The development method of automatically generating software from design models is referred to as Model-Driven Engineering. In this practice, the design of software is fully specified in a standard modelling notation before executable program code is derived [2]. Often, changes in the design model or source code will result in an instant update of the other representation. In some implementations, the software may be changed in real-time while it is running by using metamodels, specifications for how one software model maps onto a different one[8]. To further understand the field and guide conceptual development of the project, four research questions were pursued as follows.

What are the key benefits of Model-Driven Engineering? Becucci, et al. [1] report that when implemented and used well, the automatic code generation process could aid development efficiency by reducing coding time. This in turn frees up time to spend on design, simulation, validation and testing. The use of implementation-independent design models ensures that software is interoperable between systems [1]. The conversion can also avoid human error and guarantee correctly functioning code, which is especially important for embedded systems and safety-critical applications [9].

What are challenges in Model-Driven Engineering? This question was investigated to identify key areas that the project may contribute to. Clarke et al. [5] educational experience highlighted some problems in the field. The most salient is the scarcity of software modelling education in the first place, which leads to low modelling skills in graduates. When they were teaching Model-Driven Engineering to university

students, the main problems were related to the software infrastructure. They bemoaned the lack of dedicated pedagogical tools for teaching Model-Driven Engineering in particular. The majority of the students felt that the software they used (EMF), while effective, was too difficult to learn and needed more explanations, documentation and tutorials.

What are the most common design models used in industry? Gorschek et al. [4] survey of over three thousand developers showed that most respondents (76.2%) did not use any design models at all. This was followed by personal informal notation (10.9%), UML (7.9%), then all others (5%). Most software modelling was used only for communication, co-ordination or temporary brainstorming. Uptake of UML was unexpectedly low; respondents felt it was unnecessary, too big and too complicated. Even though Java is a widely used object-oriented program and can map directly to UML class diagrams, usage of UML was still low amongst Java developers too.

What other solutions exist? In order to gauge the benefits and drawbacks of current solutions, thirty-five programs and plug-ins were informally investigated by looking at their documentation and reviews. Out of those, eight were identified as free, open source and possibly extendible – Astah, AnyCode, Eclipse Epsilon, Open ModelSphere, RISE Editor, Sirius, TASTE and Yakindu Statechart. Each of these eight were personally installed and evaluated. The research and experience with these programs showed that most common problems were related to usability and convenience. Many could not be installed because of dependency issues or lack of installation instructions. Those that could be installed tended to have very complex interfaces, a scarcity of basic tutorials, vague error messages and no helpful documentation. Some even required learning a specialised language to use.

The general conclusion was that Model-Driven Engineering had intrinsic advantages but adoption was too low for industry to benefit from these. Developers generally do not see the benefits of software modelling, which may arise from a good introduction at the educational level. For the development of the project itself, Astah was identified as the easiest automatic code generation software to install, use and extend.

III. TECHNICAL DESIGN

A. Project Scope

After research and discussion, the project scope was finalised as an educational plug-in to teach Object-Oriented Programming using automatic code generation with UML and Java, at the earliest stage of programming education. This was chosen for three main reasons:

Firstly, the consistent use of design models and UML in industry is very low. This has been attributed to a lack of good educational tools that can persuasively teach the usage and importance of software design modelling [5]. The evaluation of existing solutions revealed that they are only suited for those who are already deeply familiar with the practice of model-driven development, and do not cater for beginners.

Secondly, the usage of software modelling is very low for Java developers in particular, despite the potential benefits [4].

Teaching UML with Java as a well-known reference language may help users to understand the simple link between Java code constructs and the design model elements.

Thirdly, personal experience suggests that students find Object-Oriented Programming difficult to conceptualise when introduced. This may be remedied by starting Object-Oriented Programming education with the higher level of abstraction found in visual design models, which Starett [6] showed was feasible as early as high school.

B. Software Used

The project was developed in the form of a plug-in for an existing application. This was decided to avoid “reinventing the wheel”, especially with a constrained time frame. The selected code generation application was Astah, because of the following factors identified in the solution investigations:

- Astah is free to download, meaning it is easy to obtain for the developers and users.
- Astah is compatible with all three major operating system families – Windows, Mac OS X and Linux.
- Astah is easy to install as a plug-in on any platform. This process requires only a single drag-and-drop operation.
- Astah is open source and has a free, fully documented API. This makes it possible to create new extensions of the application.
- Astah’s API supports a myriad of potential features and data queries.

C. Key Features

Rivera-Lopez, et al. [7] outlined a successful educational methodology in teaching programming through design model, which formed the basis of the project. The rough steps were, in this order: 1) Design a class diagram from a description of the problem, 2) Design a sequence diagram to determine messages between the objects, 3) Design an activity diagram to specify the internal logic of objects’ methods, 4) Design Java class code from the class diagram, 5) Design Java Main class from method calls in the sequence diagram, and 6) Design method based on activity diagrams. The student becomes aware of the simple, one-to-one relationship between the visual and textual languages.

Our project supports this educational approach by implementing a software tool for realising these steps. The UML diagrams and equivalent Java code are displayed simultaneously. The code automatically updates upon each change to the diagrams, in real-time. For ease of understanding, elements and changes in both views can be colour coded to visually establish the links between design model diagrams and source code. Constant feedback is available to guide the user. At the end, if the diagrams have been constructed correctly, the generated program can be compiled and executed. These desired features were all deemed to be possible with the Astah API which allows for access to diagram details, editing of diagrams and standard Java Swing components.

Throughout, it helps learners take their first steps by avoiding complex terminology (e.g. “polymorphism”) or complex

programming concepts. The interface complexity is also restricted to a minimum to avoid overwhelming or confusing new learners. The plug-in is not intended to function as a fully self-contained educational experience. It is fundamentally a flexible tool to be used in conjunction with customisable teacher exercises. It includes a simple tutorial that explains the usage, for any interested readers who wish to try it out¹.

IV. IMPLEMENTATION

A. Tool Overview

The tool development used Windows Command Prompt to build and launch the base Astah application, Eclipse IDE to code the plug-in itself and Github for back-ups and collaboration. The regular Astah interface includes a main diagram panel in the centre, a project panel on the left and a plug-in panel at the bottom. The project panel features a list of UML diagrams in the open project. Each one can be clicked to show the diagram in the main diagram panel, where they can be created, edited and deleted with reference to the underlying software model.

The implemented tool consists in a plug-in for Astah, as shown in Figure 2. It allows the automatic generation of Java code from UML diagrams, i.e., class, sequence and activity. The interface appears inside the plug-in panel at the bottom, which contains the interface of any loaded plug-in. The majority of the space is taken up by the text of the generated code. One class is shown at a time, and the user can navigate between classes using tabs. On the right is a small feedback section which lists errors to the user. In the corner is a button to compile and execute the code in a pop-up window, another to view the tutorial and another to view information about the plug-in itself. The plug-in automatically generates code from three types of UML diagram in the project – class diagrams for the class skeleton code, activity diagrams for the method contents (including if-branches and while-loops) and sequence diagrams for method calls between classes.

The code is generated as soon as the diagram is modified, providing the user with a fast, real time learning experience. Each Java file is represented by a tab displaying the class name. This allows for a clear and easy way to switch between files. The tool helps the user’s understanding of the code by matching code and diagrams: when selecting an element in a diagram, the corresponding line of code is highlighted in red. Conversely, when clicking on a line of code, the corresponding diagram element gets selected. If the relevant file or diagram is closed, it is automatically opened in order to facilitate the transition. The code updates in real-time, i.e. every time something in the diagram changes.

At all times, any errors in the diagrams will be described in the feedback section, e.g. if an activity diagram is missing a final node, if a sequence diagram contains a call to a non-existent method, or if a class diagram contains an attribute with the reserved Java keyword “if”. After creating a project using

¹The developed tool, namely - Lorini, is available online for reviewing at <https://briancham1994.wordpress.com/portfolio/lorini/>.

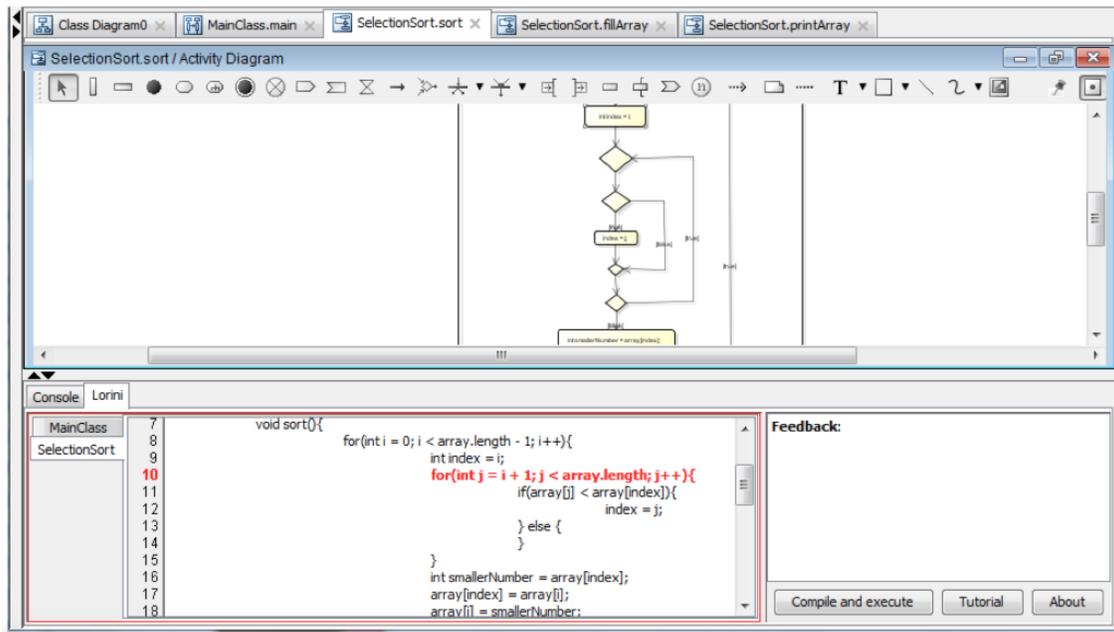


Fig. 2. Graphical User Interface of the Lorini Tool

UML diagrams, the user can click on a button in a corner to compile and execute the generated code, and see the results or compilation errors if any.

A feedback panel on the left hand side of the code panel displays explanations about the errors made by the user, helping the user understanding their mistakes and correcting them. At the bottom of this feedback panel, a button is used to launch the compilation of the Java code. Upon compilation, the output of the program is displayed, or if need be, the compilation errors arisen.

B. Technical Details

The code generation was implemented by retrieving information contained in the UML diagrams using the Astah's API and editing the diagrams as needed. In the case of class diagrams, the class name, attributes and methods are obtained and stored in string format. Each one in the open project is retrieved as an "IClassDiagram" object using the Astah API. These contain references to its name, attributes, methods, superclasses, subclasses and more. The details of these elements were extracted and put into the right locations in a String along with necessary brackets and tabbing. Each String represented the code contents of each class, and they were displayed in ScrollPanels. These were contained within a TabbedPane which allows users to switch between each class by clicking on a tab. These interface elements use regular Java Swing components, though the ScrollPanels have been extended to allow for line numbers.

In a sequence diagram, lifelines are the graphic representations of each instance of a class and messages correspond to method calls. The main method is generated from a sequence diagram by analysing the messages sent between the lifelines.

The tool checks that an instance of the class has been created before calling its methods, and displays an error message if needed. In the case of synchronous messages, corresponding to non-void method calls, an error message is displayed if no return message is present. Sequence diagrams having been restricted to depicting the main method, an error is also displayed if a lifeline other than the class containing the main method is sending a message to another lifeline, i.e., trying to call a method from another class. Similarly, the following situations will result in an error being displayed:

- Multiple sequence diagrams created.
- Name of the diagram not following the convention "class.method".
- Lifeline missing for the main class.
- Multiple lifelines created for the main class.
- Invalid lifeline created.

The activity diagram was harder to convert to Java code because of the non-linear nature of the code structure, which can include if-else statements and loops. The first step was to retrieve the correct order of the statements, which was done by following the "flows", i.e. the arrows linking each node to the next, and storing the nodes in a tree. Then, each statement was extracted and stored to be printed. However it was necessary to insert lines of code such as "else {" or closing brackets "}" as well as re-ordering the if-else statement by inserting the "if" part before the "else". This was done by analysing the incoming and outgoing flows and looking for the "true" and "false" guards. Extra care had to be taken in the cases where one of the branches was empty.

One of the main difficulties was to be able to differentiate between a loop and an if-else statement, given that they both use the true and false guards in the exact opposite manner,

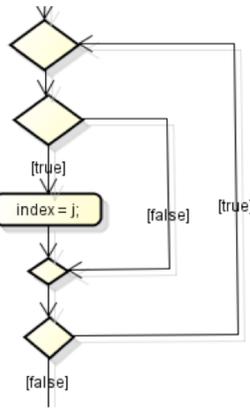


Fig. 3. Example of Loop Containing If Statement

as shown in Figure 3. Error checking had to be implemented to detect errors such as secluded nodes, if-else statements or loops missing a “true” or “false” guard and incorrect use of node types. Those errors are displayed in the feedback panel in order for the user to acknowledge and correct them, improving their learning experience.

Another example is the feedback section on the right which tells the user if there is anything wrong. This is implemented with an extended JScrollbar called “ErrorPane”. It is highly encapsulated and any other class can add an error message with only a reference to the ErrorPane object, and without having to deal with formatting. It simply has to call the method “error (String s)” with the message to display, and ErrorPane will automatically append it to the end of a single formatted bullet list.

V. EVALUATION

In order to evaluate the quality of the implemented tool, two types of evaluation methodologies were carried out, i.e., comparison with existing tools and user evaluation.

A. Tool Comparison

The implemented tool was compared to existing tools against the following criteria:

- Compatibility and portability
- Generating skeleton code from class diagrams
- Generating method code from activity diagrams
- Generating interaction code from sequence diagrams
- Code and diagram matching
- Instantaneous code generation
- Code compilation and execution
- Generating code in multiple programming languages
- Generating code from design models directly and automatically, without requiring the user to learn a dedicated language, diagram or syntax.

The implemented tool was found to be competitive on many aspects. Firstly, being portable and free for students, it is very accessible. It also benefits from an intuitive interface and uses only common standards such as Java and UML, avoiding the

overhead of having to learn a specific language or diagram syntax. The educational aspect of the tool makes it unique compared to its competitors, allowing users to match diagrams and code with the help of highlighting and instantaneous code generation. Finally, the availability of feedback and compilation tools yields a better understanding of the process.

Partial comparison results² against a set of common UML tools can be found in Figure 4. Overall, the comparison highlighted the factors that make the project unique amongst similar tools:

- **Convenient** – It is free of charge and compatible on all platforms.
- **Intuitive to use** – It does not require learning any special syntax or diagrams to perform the code generation. It uses standard UML and Java, not a dedicated language or format.
- **Educational** – It performs automatic highlighting of match between code and diagram, and vice-versa, which constantly update to reflect each other (i.e. on-the-fly code generation). This allows users to understand and explore at their own pace. Other tools assume that the user is already familiar with this relationship and does not help them to learn it. The ones that include these features either have limited functionality, or use their own formats instead of standards like UML and Java, limiting their educational use.
- **Feedback** – It tells the user basic details of anything wrong with the software model. It can also compile and check results of the code.

B. User Evaluation

User testing was performed in order to get a first-hand feedback on the implemented tool. The eight volunteers were aged 18 to 25 and included both males and females, experienced and neophytes in terms of programming and UML design. One of them had previous experience with Astah. Participants were asked to perform some basic tasks with the plug-in such as designing a “Hello World” program using the three supported types of UML diagrams. They were presented with a set of Likert-scale based questions[10]. For an easier analysis, the Likert items were converted to a numerical scale, 1 corresponding to “strongly disagree” and 5 corresponding to “strongly agree”. The results were then averaged and are presented in Figure 5.

In average, the plug-in was considered very intuitive and uncluttered. Users found the code-diagram matching useful in order to understand the design process. However, some bugs were uncovered and some of the feedback messages were judged to be ambiguous and confusing. Moreover, the need for a tutorial was noticed. Following the evaluation results, improvements were made to the plug-in. A full testing session was carried out to identify and fix a plethora of remaining bugs. All problematic feedback messages were rewritten until users found them clearer. Thorough testing was performed and

²More detailed tool comparisons are available on the Lorini web page.

Tool	Compatibility			Code generation features							
	Windows	Mac	Linux	Generates class skeleton code	Generates method code	Generates class interaction code	Code/diagram matching	On-the-fly code generation	Code compilation/execution	Multiple programming languages	Code conversion syntax not required
This project	Y	Y	Y	Y	Y	Y	Y	Y	Y	N	Y
Astah Professional	Y	Y	Y	Y	N	N	N	N	N	Y	Y
ArgoUML	Y	Y	Y	Y	N	N	N	N	N	Y	Y
EMF	Y	Y	Y	Y	N	N	N	N	N	N	N
Maple	Y	Y	Y	N	Y	N	N	N	Y	Y	N
SimuLink	Y	Y	Y	Y	Y	Y	Y	N	Y	Y	N

Fig. 4. Tool Comparison Results

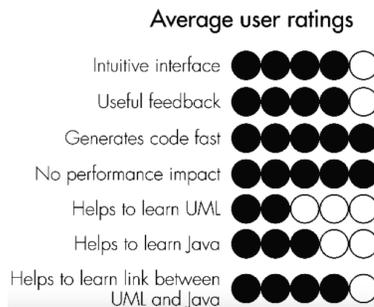


Fig. 5. User Evaluation Results

all uncovered bugs were fixed. The full tutorial was embedded into the plug-in to help with the understanding of the tool.

VI. CONCLUSION

While Model Driven Engineering is proven to be time-efficient and less prone to human errors, research showed that design models are scarcely used, mostly due to lack of education and training in the matter. In parallel, it has been shown that teaching design modelling to students before programming is feasible and even beneficial. These formed the basis of an educational approach that uses automatic code generation to teach Object-Oriented Programming and its relationship with design models. The implemented tool is a plug-in for Astah able to automatically generate Java code from three types of UML diagrams, namely activity, sequence and class diagrams. The tool includes the following features:

- Instantaneous code generation
- Code-diagram matching
- Feedback panel displaying user errors
- Compilation and execution of the code

Evaluation of the tool indicated it to be competitive in many aspects, mainly due to its unique educational features, as well as being easy to install and very intuitive to use.

In the future, we plan to conduct a large scaled user evaluation and focus more on measuring the education aspects of the developed tool. On the technical side, we would like to extend the current implementation to support the code generation in different programming languages, and from other types of UML diagrams.

REFERENCES

- [1] M. Becucci, A. Fantechi, M. Giromini, and E. Spinicci, "A comparison between handwritten and automatic generation of c code from sdl using static analysis," *Software: Practice and Experience*, vol. 35, no. 14, pp. 1317–1347, 2005.
- [2] G. Nisha, "A model driven approach for design and development of a safety critical system," in *Electronics Computer Technology (ICECT), 2011 3rd International Conference on*, vol. 4, April 2011, pp. 15–18.
- [3] H. Zhu, J. Sun, J. S. Dong, and S.-W. Lin, "From verified model to executable program: the pat approach," *Innovations in Systems and Software Engineering*, vol. 12, no. 1, pp. 1–26, 2015.
- [4] T. Gorschek, E. Tempero, and L. Angelis, "On the use of software design models in software development practice: An empirical investigation," *J. Syst. Softw.*, vol. 95, pp. 176–193, Sep. 2014.
- [5] B. Tekinerdogan, "Experiences in teaching a graduate course on model-driven software development," *Computer Science Education*, vol. 21, no. 4, pp. 363–387, 2011.
- [6] C. Starrett, "Teaching uml modeling before programming at the high school level," in *Advanced Learning Technologies, 2007. ICALT 2007. Seventh IEEE International Conference on*, July 2007, pp. 713–714.
- [7] R. Rivera-Lopez, E. Rivera-Lopez, and A. Rodriguez-Leon, "Another approach for the teaching of the foundations of programming using UML and Java," in *Proceedings of the 3rd WSEAS International Conference on Computer Engineering and Applications*, ser. CEA'09. Stevens Point, Wisconsin, USA: World Scientific and Engineering Academy and Society (WSEAS), 2009, pp. 279–283.
- [8] F. Krichen, B. Hamid, B. Zalila, M. Jmaiel, and B. Coulette, "Development of reconfigurable distributed embedded systems with a model-driven approach," *Concurrency and Computation: Practice and Experience*, vol. 27, no. 6, pp. 1391–1411, 2015.
- [9] M. Hinchey, J. Rash, and C. Rouff, "Requirements to design to code: Towards a fully formal approach to automatic code generation," Technical Report TM-2005-212774, NASA Goddard Space Flight Center, Greenbelt, MD, USA, Tech. Rep., 2004.
- [10] O. Laitenberger and H. M. Dreyer, "Evaluating the usefulness and the ease of use of a web-based inspection data collection tool," in *Software Metrics Symposium, 1998. Metrics 1998. Proceedings. Fifth International*, Nov 1998, pp. 122–132.

SLTM: A Sentence Level Topic Model for Analysis of Online Reviews

Yuhan Zhang and Haiping Xu

Computer and Information Science Department
University of Massachusetts Dartmouth, Dartmouth, MA 02747, USA
{yzhang5, hxu}@umassd.edu

Abstract—Due to large amounts of reviews for many similar online products, users often feel difficult to determine which products have the most desirable features that they want. In this paper, we propose a model-based approach to analyzing online reviews and identifying the strengths and weaknesses of a product by its product features. We propose a Sentence Level Topic Model (SLTM), which can classify review sentences into different classes corresponding to different product features. The model contains a hidden layer, called the topic layer, between corpus and words. Once a SLTM has been trained with sufficient labeled data points, it can identify the most related topic (i.e., product feature) for each sentence. To capture a reviewer’s opinion, we perform sentiment analysis for each review sentence, and derive the weighted feature preference vectors for the review. Finally, we combine the results of all review comments for a product into a review summary. The case study shows that by comparing the review summaries of similar online products, users may have a much easier time to find their desired products.

Keywords—Electronic commerce; product review; product feature; topic model; feature extraction; sentiment analysis.

I. INTRODUCTION

Many e-commerce websites adopt the average star rating mechanism to help customers with their buying decisions; however, such ratings are not accurate and do not necessarily reflect the actual quality of the products [1]. To deal with this issue, major e-commerce websites allow users to provide reviews for the products they bought. A popular online product listed at e-commerce websites such as Amazon, often receives hundreds or even thousands of reviews. Due to the large amount of reviews, customers often have a hard time to read all of them, and tend to miss important product information. To provide a better view about the products and save customers’ time for browsing the reviews, it is necessary to provide an approach that can automatically analyze all reviews of a product and output its strengths and weaknesses in terms of its product features. In this paper, we propose a Sentence Level Topic Model (SLTM), which can be used to classify sentences into different topics, where each topic refers to a product feature (in this paper, we will use the terms “topic” and “feature” interchangeably). The SLTM introduces a hidden layer, called the topic layer, between corpus and words. By applying Bayes’ rule, SLTM can effectively identify the product feature in each sentence. When a feature is identified, we perform sentiment analysis of the sentence to find how the reviewer likes or dislikes the feature. When all

review comments have been processed, we combine the results into a review summary. To illustrate the feasibility and effectiveness of our approach, we retrieved and processed review comments for a number of similar products from Amazon. We show that by comparing the strengths and weaknesses of product features among similar products, our approach can greatly save customers’ time for making decisions on selecting the most desired online products.

In the past decades, review mining has attracted a great deal of attention due to the rapid growth of online reviews. Pang and Lee employed three machine-learning approaches to label the polarity of IMDb movie reviews [2]. They proposed to first extract the subjective portion of text with a graph min-cut algorithm, and then feed them into a sentiment classifier. Rather than applying the straightforward frequency-based bag-of-words feature selection methods, Whitelaw *et al.* defined the concept of “adjectival appraisal groups” headed by an appraising adjective and optionally modified by words like “not” or “very” [3]. Each appraisal group was further assigned four types of features: attitude, orientation, graduation, and polarity. Different from the above approaches, our SLTM method is a statistical approach using topic modeling, which can be directly applied at sentence level to maximize the accuracy for text mining.

In machine learning and Natural Language Processing (NLP), a topic model is defined as a type of statistical model for discovering the abstract “topics” that occur in a collection of documents. An early topic model for text mining, called Latent Semantic Indexing (LSI), proposed by Papadimitriou *et al.*, is an information retrieval technique based on the spectral analysis of the term-document matrix [4]. They showed that under certain conditions, LSI could capture the underlying semantics of the corpus and achieve improved retrieval performance. Blei *et al.* developed a very commonly used topic model, called Latent Dirichlet Allocation (LDA) [5], which is a three-level hierarchical Bayesian model that allows documents to have a mixture of topics. Different from LSI and LDA, since one sentence in a review typically contains at most one topic, our model only requires one matrix, namely the topic-word matrix. Therefore, SLTM is a one-layer classifier, which could be more suitable for analysis of online reviews.

Symbolic techniques are one of the major approaches to detecting sentiment from text [6]. Symbolic techniques assume the corpus is a “bag of words”; therefore, the relationships between the individual words are not considered. Kamps *et al.* used the lexical database WordNet to determine the emotional content of a word along different dimensions [7]. WordNet is a

large lexical database of words containing nouns, verbs, adjectives and adverbs, which are grouped into sets of cognitive synonyms (synsets) that describe different concepts. Based on WordNet, SentiWordNet has been developed as an extended lexical resource for opinion mining [8]. To support sentiment analysis, SentiWordNet assigns each synset of WordNet three sentiment scores, namely positivity, negativity and objectivity. In this paper, we adopt a similar tool called the Stanford CoreNLP toolkit, which supports most of the common core NLP functions, including Part-Of-Speech (POS) tagger and sentiment analysis [9].

II. TOPIC MODEL BASED REVIEW ANALYSIS

A. A Framework for Sentence Level Topic Modeling

The framework for sentence level topic modeling of online product reviews consists of four major parts, namely review preprocessing, feature extraction, sentiment analysis, and product summary. As shown in Fig. 1, the system is flexible and highly modularized, which supports processing a collection of reviews for a certain online product. The reviews can be processed either sequentially or in parallel as we consider them being written by independent customers.

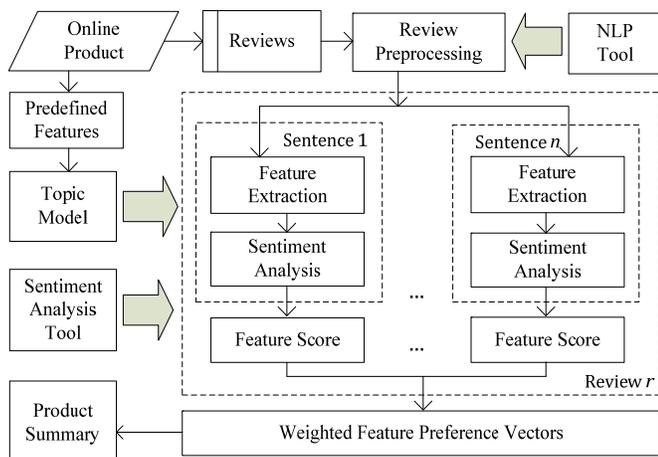


Figure 1. A framework for sentence level topic modeling

To identify product features and analyze sentiment, we first collect the reviews for a class of online products P (e.g. camera). Then we define a set of product features for P . As shown in Fig. 1, the input of the framework is a collection of reviews for product $p \in P$, and the output is a product summary that specifies the weighted feature preference for each predefined product feature. In the review preprocessing module, a review is split into a number of sentences that are parsed and tagged using an NLP tool. The tagged sentences are then sent to the feature extraction modules, where we use SLTM to calculate the probability of each feature. If a feature is identified in a sentence, the feature extraction module outputs the sentence to the sentiment analysis module. The sentiment analysis module then calculates a feature score for the sentence as an integer between 0-4, where 0 representing very negative and 4 representing very positive.

Once we calculated the weighted feature scores for each review r , we combine them in feature preference vectors, and

derive the product summary that shows how strongly each of the product features is supported or not supported by buyers.

B. Review Preprocessing

The review preprocessing module processes a single review each time. Using Stanford CoreNLP sentence split toolkit, a review is split into a number of sentences. During this process, sentences with less than two words are ignored since a meaningful sentence should at least have a topic word and a sentiment word. Then we use Stanford CoreNLP POS tagger to indicate each word in the sentence as noun, verb, adjective, adverb or other tags. We use the following sentence as an example to show how POS tagger parses a sentence:

“The display is nice and big.”

Using the CoreNLP toolkit, the parser outputs the sentence with the following POS tags:

DT NN VBZ JJ CC JJ
The display is nice and big.

In the above tagged sentence, the tags DT, NN, VBZ, JJ, and CC stand for “Determiner”, “Noun”, “Verb, third person singular present”, “Adjective”, “Coordinating Conjunction”, respectively. After detecting all sentences in a review and parsing each sentence with POS tags, we save each sentence along with the POS tag information into a local file, which will be used to identify product features as described in the following sections.

Based on our observation, a topic in a sentence is usually determined by nouns or adjectives. Thus we can simplify the model by using POS tags to get rid of those noise data (words such as “You”, “I”, “a”, “the”, etc.). As in LSA, we consider a sentence a “bag of words”. Therefore, we output only the words with tag NN or JJ to the feature extraction module for further processing. In addition, for each word, we consider the lemma only, which is the dictionary form of a set of words. For example, words “likes”, “like”, “liked” are forms of the same lexeme, with “like” as the lemma.

III. FEATURE IDENTIFICATION USING SLTM

A. Topic-Word Matrix

To develop a topic model for product type P , we predefine a list of product features based on customers’ interests as well as product specification. The predefined features are considered different topics in our topic model. Suppose we have defined K features for P . We create a *topic-word* matrix Φ with K rows and V columns, where V is a dynamic number that increases by 1 each time a new word appears. Initially, Φ has K rows and 0 column, which is an empty matrix due to the emptiness of the initial vocabulary list. After the matrix has been expanded, each row of the matrix represents the distribution of words in the corresponding topic, while each column indicates how the corresponding word has co-occurred with different topics. The elements of Φ are defined as follows.

Φ_k : distribution of words in topic k (k ranges from $0 \sim K-1$)

$\Phi[k, w]$: counts of word w occurring in topic k (w ranges from $0 \sim V-1$)

The matrix is used to record topic-word co-occurrence information with each cell initially set to 0. To expand the

column (vocabulary) in matrix Φ and increase the number in each cell of Φ , we first manually create a training dataset with labeled sentences as follows.

For a review sentence, we tag it with proper product feature if it contains one. For example, the sentence “The display is nice and big.” describes the product feature “ViewScreen”; therefore, we put a “#ViewScreen” tag at the end of this sentence:

“The display is nice and big.” #ViewScreen

Similarly, we can label the following two review sentences in the same way:

“They had the best price on the camera, and got it here on time!” #Price

“It is also much easier to see camera settings on the big LCD.” #ViewScreen

Note that a customer may use various words or phrases to describe a feature. For example, words or phrases “display” and “big LCD” all refer to the same feature “ViewScreen”.

Since only nouns and adjectives are considered inputs of the topic model, each labeled data point is defined as a bag of nouns and adjectives with its associated feature tag. For example, the data points for the aforementioned three tagged sentences are recorded as follows:

“display, nice, big” #ViewScreen

“best, price, camera, time” #Price

“easy, camera, setting, big, LCD” #ViewScreen

The procedure for setting up the topic-word matrix Φ is presented as Algorithm 1. In this algorithm, we first initialize Φ with the vocabulary list vl being empty. Then for each data point d , let f be the feature (tag) of d , and $topicIndex$ be the index of f in the topic list tl . For each word w in d , check if w appears in vl . If w is in vl , let $wordIndex$ be the index of w ; otherwise, add w to vl and expand Φ by one column for the new word w , and let $wordIndex$ be the index of the last column of the matrix. Finally, the topic-word co-occurrence count $\Phi[topicIndex][wordIndex]$ is increased by one. Note that the topic-word matrix Φ can be updated in the same way when more labeled data points become available.

Algorithm 1: Set up Topic-Word Matrix Φ

Input: Matrix Φ , topic list tl , vocabulary list vl , and a training dataset

Output: updated matrix Φ

1. Initialize matrix Φ with vl as an empty list
 2. **for each** data point d from the training dataset
 3. let f be the feature of d , and $topicIndex$ be the index of f in tl
 4. **for each** word w in d
 5. **if** vl contains w
 6. let $wordIndex$ be the index of w in vl
 7. **else**
 8. add w to vl , add 1 column to Φ , and set $wordIndex = |vl| - 1$
 9. increase $\Phi[topicIndex][wordIndex]$ by 1
 10. **return** matrix Φ
-

B. Feature Identification

Once we have updated the topic-word matrix using all data points from the training dataset, we can use it as a discriminative model to classify sentences into different class.

Let T_0, \dots, T_{K-1} be the list of topics, and $P(T_k)$, where $0 \leq k \leq K-1$, be the probability that topic T_k appears in a sentence. The probability $P(T_k)$ can be calculated as in Eq. (1).

$$P(T_k) = \frac{N(T_k)}{\sum_{i=0}^{K-1} N(T_i)} \quad (1)$$

where $N(T_k)$ is the number of times that topic T_k has appeared in the training dataset, and $\sum_{i=0}^{K-1} N(T_i)$ is the total number of data points in the training dataset. Obviously, $\sum_{k=0}^{K-1} P(T_k) = 1$.

Let $P(T_k | S)$ be the probability that topic T_k appears given sentence S , where $0 \leq k \leq K-1$, and $\sum_{k=0}^{K-1} P(T_k | S) = 1$. Assume each sentence has at most one topic, and the topic of sentence S $Topic_S$ can be determined by Eq. (2).

$$Topic_S = \begin{cases} T_m & \text{if } P(T_m | S) > 0.5 \\ Null & \text{otherwise} \end{cases} \quad (2)$$

where $P(T_m | S) = \max(P(T_1 | S), P(T_2 | S), \dots, P(T_K | S))$

Note that in Eq. (2), 0.5 is the threshold for topic identification. If there is a tie for determining T_m , none of the topics satisfies the condition $P(T_k | S) > 0.5$. In this case, the topic of sentence S is determined as “Null”.

Now consider the calculation of probability of sentence S having topic T_k . By the Bayes’ rule, $P(T_k | S)$ can be calculated as in Eq. (3.1).

$$P(T_k | S) = \frac{P(S | T_k) * P(T_k)}{P(S)} \quad (3.1)$$

As we treat S as a “bag of words”, where each word has a unigram meaning, to simplify our model, we consider the words bring independent from each other. Let S be $\{w_1, \dots, w_n\}$. Eq. (3.1) can be rewritten as in Eq. (3.2).

$$P(T_k | S) = \frac{P(w_1, \dots, w_n | T_k) * P(T_k)}{P(w_1, \dots, w_n)} \quad (3.2)$$

where $P(w_1, \dots, w_n | T_k)$ and $P(w_1, \dots, w_n)$ are defined as in Eq. (4.1-4.2).

$$P(w_1, \dots, w_n | T_k) = \prod_{i=1}^n P(w_i | T_k) \quad (4.1)$$

$$P(w_1, \dots, w_n) = \prod_{i=1}^n P(w_i) \quad (4.2)$$

Note that $P(w_i | T_k)$, where $1 \leq i \leq n$, is the probability that word w_i appears in a sentence given topic T_k . Since Φ_k is the distribution of words in topic T_k , $P(w_i | T_k)$ can be calculated as in Eq. (5).

$$P(w_i | T_k) = \frac{\Phi[k, i]}{\sum_{l=0}^{V-1} \Phi[k, l]}, \text{ where } 1 \leq i \leq n \quad (5)$$

where $\sum_{l=0}^{V-1} \Phi[k, l]$ is the sum of word counts in topic T_k . Since each $P(w_i | T_k)$ could be a very small decimal, to avoid accuracy overflow errors in calculating $P(w_1, \dots, w_n | T_k)$, we first calculate the logarithm of $P(w_1, \dots, w_n | T_k)$ as in Eq. (6.1), and then derive $P(w_1, \dots, w_n | T_k)$ as in Eq. (6.2).

$$\ln(P(w_1, \dots, w_n | T_k)) = \sum_{i=1}^n (\ln(\Phi[k, i]) - \ln(\sum_{l=0}^{V-1} \Phi[k, l])) \quad (6.1)$$

$$P(w_1, \dots, w_n | T_k) = e^{\ln(P(w_1, \dots, w_n | T_k))} \quad (6.2)$$

Finally, $P(w_i)$, where $1 \leq i \leq n$, is the probability that word w_i appears in a sentence. $P(w_i)$ can also be calculated using the topic-word matrix; however, the calculation of $P(w_i)$ is not necessary, as it is a constant for all topics, so is $P(w_1, w_1, \dots, w_n)$. Since $\sum_{k=0}^{K-1} P(T_k | S) = 1$, we can calculate $P'(T_k | S)$ as in Eq. (7.1), and then normalize it into $P(T_k | S)$ using Eq. (7.2).

$$P'(T_k | S) = P(w_1, \dots, w_n | T_k) * P(T_k) \quad (7.1)$$

$$P(T_k | S) = \frac{P'(T_k | S)}{\sum_{i=0}^{K-1} P'(T_k | S)} \quad (7.2)$$

IV. SENTIMENT ANALYSIS AND REVIEW SUMMARY

A. Sentiment Analysis

Once we have extracted product features from review sentences, the next task is to analyze the customer's preference of the extracted feature. Here we utilize Stanford Sentiment Analysis toolkit for this purpose. Stanford Sentiment Analysis model uses a type of Recursive Neural Network (RNN) based on the grammatical structure of a sentence. The training data set of this model is from Stanford Sentiment Treebank, and it is worth mentioning that users can help to improve this model while using it. The toolkit first split the sentence into phrases, then phrases into words. For the sentiment calculation, the toolkit uses bottom up algorithm, which calculates the sentiment for each word first, then for each phrase, and eventually for the whole sentence. Since we assume one sentence contains only one topic, the sentiment score of whole sentence is also the score for the topic. According to reference [10], the model's accuracy on a single sentence classification is above 80%, while the accuracy of predicting fine-grained sentiment for all phrases could be even higher.

B. Review Summary

Our approach is to extract features and calculate sentiment scores at the sentence level. Once we have recorded the feature score for each sentence, we can summarize the user preferences of the predefined product features for a certain product. The summarization is based on the procedure for combining weighted feature preference scores, which is described as in Algorithm 2.

Algorithm 2: Feature Preference Score Combination

Input: Feature scores of all reviews for product p

Output: Vectors LIKE and DISLIKE indicating how customers like or dislike product p , respectively, in terms of the product features.

1. initialize two K -dimension vectors LIKE and DISLIKE to 0.
 2. **for each** review r for product p
 3. initialize K -dimension vector PREFER to 0.
 4. **for each** sentence s in r
 5. let f be the feature in s , and $tIndex$ be the index of feature f in the vectors; let $sentiScore$ be the sentiment score of s
 6. **switch** $sentiScore$
 7. **case** 0: PREFER[$tIndex$] = -1; break; // very negative
 8. **case** 1: PREFER[$tIndex$] = -0.7; break; // negative
 9. **case** 2: PREFER[$tIndex$] = 0; break; // neutral
 10. **case** 3: PREFER[$tIndex$] = 0.7; break; // positive
 11. **case** 4: PREFER[$tIndex$] = 1; break; // very positive
-

-
12. **for each** element i in PREFER, where $0 \leq i \leq K-1$
 13. **if** PREFER[i] > 0 LIKE[i] += PREFER[i]
 14. **else** DISLIKE[i] += PREFER[i] * (-1)
 15. **return** vectors LIKE and DISLIKE
-

In Algorithm 2, all review results are combined to evaluate how customers like or dislike the predefined product features of a product. Note that if a review contains multiple review sentences related to feature f , according to the algorithm, only the last sentence given by the reviewer is considered.

V. CASE STUDY

To demonstrate the feasibility of our approach, we use an example of online products from Amazon website. The product type is camera, with listed online product such as "Nikon Coolpix L300 Digital Camera (Black)." Although the products we selected are good ones due to their high average star ratings, customers will still want to know different features of the products. For example, does the battery last long enough or is the camera small and light enough for a long trip? By analyzing the review comments, our approach may help customers to answer such questions.

A. Product Features

To evaluate the product features of digital camera, we define 9 customer-interested features as well as a "Null" feature, listed as follows.

1. **Null:** is a dummy feature that indicates a sentence having no specific topic.
2. **Lens:** is an optical device on a camera that can change the focus of a light beam through refraction. The quality of lens has a strong impact on the quality of the camera.
3. **SizeWeight:** refers to the appearance of a camera in terms of its size and/or weight.
4. **Price:** indicates whether the price of the camera is reasonable or not.
5. **Resolution:** is a feature of picture quality, and higher resolution typically indicates higher picture quality.
6. **Stabilization:** is a feature that indicates whether a camera can effectively prevent or compensate for unwanted camera movement.
7. **Accessory:** refers to the quality or availability of camera accessories, such as the quality of battery and the availability of SD card option.
8. **Shutter:** is a device associated with a camera that allows light to pass for a determined period of time.
9. **ViewScreen:** also known as LCD or viewfinder, which is a device used to display images.
10. **Mode:** refers to manual mode or automatic mode that can be used in various situations.

Note that the "Null" feature is a dummy one indicating that a sentence does not describe any feature of the camera. For example, sentences such as "I love this camera so much," "My last camera was the Canon Powershot A495," or "So happy I got this," do not make comments on any product feature. Instead, they simply either state a fact or express a general feeling about the camera.

B. Experimental Results

We trained the SLTM using about 1500 labeled data points and achieved 95% accuracy. In the experiments, we chose four different digital cameras sold at Amazon, and produced their review summaries. The selected four digital cameras are listed in Table 1, which have similar prices around \$200 and all sold by top-100 camera sellers at Amazon. Moreover, they all have above 4.0 average star ratings; thus, it is hard for a buyer to determine which one is the most suitable one to buy.

Table 1. Four different digital cameras listed at Amazon

Product Name	ASIN	Star Ratings	Price	Reviews
Nikon Coolpix	B00HQDBLDO	4.3	\$165	384
Canon PowerShot SX520	B00M0QVTOS	4.4	\$269	315
Canon Rebel XT (Used)	B0007QKN22	4.1	\$205	648
Canon PowerShot SX400	B00M0QVG3W	4.3	\$150	364

The review summaries for comparing the four selected cameras are presented in Fig. 2. In the figure, the left-side bars indicate the weighted DISLIKE scores for each product feature; while the right-side bars indicate the weighted LIKE scores for the features. From the figure, we can see that Nikon Coolpix has a really bargain price, perfect size & weight; however, its view screen, battery and lens are not good enough. Cannon SX520's price is also great, it has a good resolution and battery, and its size & weight is acceptable; however, its shutter and view screen are all not satisfactory. Cannon Rebel XT's price, setup, shutter, battery and lens get more complains than the other two, and thus does not look like a good deal. Finally, Cannon SX400 has acceptable price and size & weight, but its setup, view screen, shutter, battery, and lens are its weaknesses. Among the four products, Nikon Coolpix and Cannon SX520 seem to be more desirable products than the other two because they have more strengths than weaknesses in terms of their product features.

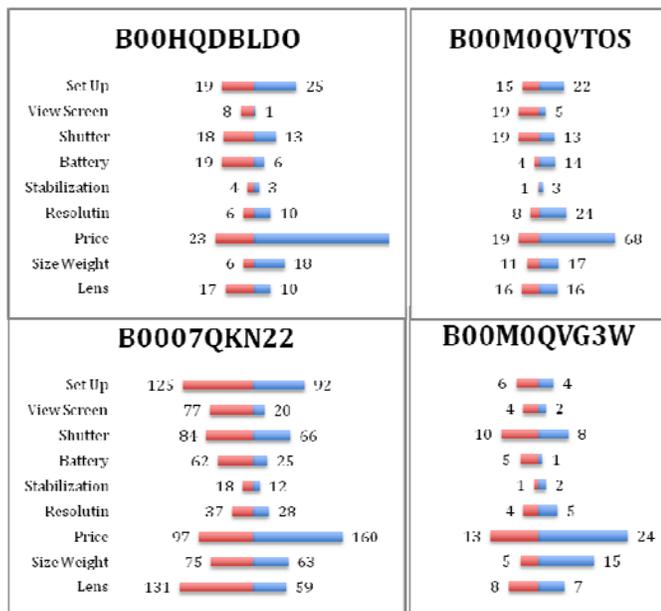


Figure 2. Review summaries of four selected cameras

VI. CONCLUSIONS AND FUTURE WORK

In this paper, we introduced a sentence level topic model for evaluating and comparing online products based on their reviews. In our approach, we use product reviews as pieces of evidence to identify the strengths and weaknesses of a product in terms of its product features. To illustrate the feasibility and effectiveness of our approach, we retrieved product information of online products and their review comments from Amazon for review analysis. Our case study shows that our approach can achieve high accuracy with a training dataset of a reasonable size. As a major benefit of our approach, customers can greatly save their time for reading reviews and comparing similar products based on their product features.

In future research, we plan to develop our own sentiment analysis tool to enhance system performance. We will also study the impacts on performance when considering a sentence as a bag of independent words vs. related words. We will develop useful tools to allow users to help with labeling data points. Finally, we will further validate the feasibility of our approach using labeled datasets from different domains.

REFERENCES

- [1] R. Wei and H. Xu, "A Formal Cost-Effectiveness Analysis Model for Product Evaluation in E-Commerce," In *Proceedings of the 25th International Conference on Software Engineering and Knowledge Engineering (SEKE 2013)*, Boston, MA, USA, June 27-29, 2013, pp. 287-293.
- [2] B. Pang and L. Lee, "A Sentimental Education: Sentiment Analysis Using Subjectivity Summarization Based on Minimum Cuts," In *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics (ACL)*, 2004, pp. 271-278.
- [3] C. Whitelaw, N. Garg, and S. Argamon, "Using Appraisal Groups for Sentiment Analysis," In *Proceedings of the 14th ACM International Conference on Information and Knowledge Management (CIKM)*, 2005, pp. 625-631.
- [4] C. H. Papadimitriou, P. Raghavan, H. Tamaki, and S. Vempala, "Latent Semantic Indexing: A Probabilistic Analysis," *Journal of Computer and System Sciences*, Vol. 61, No. 2, October 2000, pp. 217-235.
- [5] D. M. Blei, A. Y. Ng, M. I. Jordan, "Latent Dirichlet Allocation," *Journal of Machine Learning Research*, Vol. 3, January 2003, pp. 993-1022.
- [6] E. Boiy, P. Hens, K. Deschacht, and M.-F. Moens, "Automatic Sentiment Analysis in Online Text," In *Proceedings of the 11th International Conference on Electronic Publishing (ELPUB 2007)*, Vienna, Austria, June 13-15, 2007, pp. 349-360.
- [7] J. Kamps, M. Marx, R. J. Mokken, and M. De Rijke, "Using WordNet to Measure Semantic Orientations of Adjectives," In *Proceedings of the 4th International Conference on Language Resources and Evaluation (LREC-04)*, May 2004, pp. 1115-1118.
- [8] A. Esuli and F. Sebastiani, "SentiWordNet: A Publicly Available Lexical Resource for Opinion Mining," In *Proceedings of the 5th International Conference on Language Resources and Evaluation (LREC-06)*, 2006, pp. 417-422.
- [9] C. D. Manning, M. Surdeanu, J. Bauer, J. Finkel, S. J. Bethard, and D. McClosky, "The Stanford CoreNLP Natural Language Processing Toolkit," In *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, Baltimore, Maryland, USA, June 2014, pp. 55-60.
- [10] Y. R. Socher, A. Perelygin, J. Y. Wu *et al.*, "Recursive Deep Models for Semantic Compositionality Over a Sentiment Treebank," In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP 2013)*, Seattle, WA, USA, October 18-21, 2013, pp. 1631-1642.

Using Frequent Closed Pattern Mining to Solve a Consensus Clustering Problem

Atheer Al-najdi

Nicolas Pasquier

Frédéric Precioso

Univ. Nice Sophia Antipolis, CNRS, I3S, UMR 7271, 06900 Sophia Antipolis, France

E-mail: {alnajdi, pasquier, preciososo}@i3s.unice.fr

Abstract

Clustering is the process of partitioning a dataset into groups based on the similarity between the instances. Many clustering algorithms were proposed, but none of them proved to provide good quality partition in all situations. Consensus clustering aims to enhance the clustering process by combining different partitions obtained from different algorithms to yield a better quality consensus solution. In this work, we propose a new consensus method that uses a pattern mining technique in order to reduce the search space from instance-based into pattern-based space. Instead of finding one solution, our method generates multiple consensus candidates based on varying the number of base clusterings considered. The different solutions are then linked and presented as a tree that gives more insight about the similarities between the instances and the different partitions in the ensemble.

keywords Unsupervised learning; Clustering; Consensus clustering; Ensemble clustering; Frequent closed itemsets.

1. Introduction

Clustering is one of the important tasks in data mining. It can discover patterns in a dataset by classifying the instances into groups based on their similarities. However, different algorithms were proposed in the last decades to enhance this unsupervised learning process. Unfortunately, none of these algorithms proved to provide “good” clustering solution in all situations. Therefore, a new branch of research emerged in the last years that focuses on combining different clusterings into one consensus solution, known as *Consensus Clustering* or *Clusterings Aggregation*. The idea behind it is that we can benefit from the advantages of each algorithm used in building the initial clusterings ensemble, to produce a new solution that is more stable and achieves better quality final result.

Different approaches were used to generate the consensus solution. However, until very recently, all these methods search for the optimal consensus in the whole search

space, making some of them inapplicable for large datasets. Thus, new research focused on finding the consensus partition in a pruned space. Wu *et. al.* [22] worked on a reduced space of “data fragments” instead of the instances space. Vega-Pons and Avesani [20] defined two functions to prune the search space, namely the unanimity and the majority prune functions. In this work, we use the *Frequent Closed Itemsets (FCI)* [14] technique from the domain of pattern mining and association rules discovery, in order to find a different pruning of the search space. FCI, a technique designed to discover patterns in very large datasets, can be used to transform the search space from instance-based into pattern-based space. Each pattern defines agreement between a set of base clusters on grouping a set of instances. Therefore, we can even partition this patterns space into subspaces based on the number of base clusterings that define the patterns. On each subspace, we find a consensus solution by clustering the patterns based on their similarity. Thus, our approach involves generating multiple consensus, then recommend the one the most similar to the ensemble. All these solutions are linked and presented in a tree of consensus clusters that enables the analyst to discover which clusters are more stable than others, pointing out strong intra-cluster similarity between the instances.

This paper is organized as follows: The next section provides a brief summary of the main categories of consensus clustering methods. The proposed approach is explained in Sect. 3. Description of the performed tests and the achieved results is presented in Sect. 4. We conclude in Sect. 5.

2. Related Work

Different methods were proposed to find a consensus partition from an ensemble of clusterings. They can be categorized based on the underlying approach used:

- **Graph partitioning methods:** By representing the relation between the instances and the base clusters they belong to as a graph, a consensus solution is obtained using a graph partitioning algorithm. For example: Cluster-based Similarity Partitioning Algorithm (CSPA), HyperGraph Partitioning Algorithm

(HGPA), Meta CLustering Algorithm (MCLA) (Strehl and Ghosh [17]), and Hybrid Bipartite Graph Formulation (HBGF) (Fern and Brodley [5]).

- **Voting methods:** These methods try first to find a unified labeling among the base clusterings, then apply a voting approach to generate the consensus solution. In the Plurality Voting method (Dudoit and Fridlyand[4], Fischer and Buhmann[6]), the relabeling problem is solved using the Hungarian algorithm, then the final label of an instance is the one the mostly assigned to it in the relabeled ensemble.
- **Co-association based methods:** A co-association matrix can be used to define the similarity between the instances in terms of how many times each pair belongs to the same cluster in the ensemble. Then, a clustering algorithm can be applied to find the consensus partition. For example, Fred and Jain [7] used the single-linkage hierarchical clustering algorithm on the matrix to generate a consensus solution.
- **Finite mixture methods:** The labels of the base clusterings are considered as random variables drawn from a probability distribution. The consensus partition here is the solution of a maximum likelihood estimation problem, using for example the EM algorithm as done by Topchy *et. al.* [18].

More details in the surveys by Ghaemi *et al.* [8], Sarumathi *et al.* [16], and Vega-Pons & Ruiz-Shulcloper [21].

3. The Proposed Pattern-Based Approach

To discover the relationships between the instances and the base clusters in the ensemble, we use the frequent closed itemset technique from pattern mining discipline. FCI finds first the sets of instances that are clustered together by all base clusterings, similar to the data fragments in [22] or the unanimity function in [20]. Instead of using the algorithms proposed in [22], or the majority function in [20] that provides further pruning to the search space, FCI finds also the sets of instances that are clustered together by different combinations of base clusterings. Therefore, FCI does not just transform the search space from instance-based into patterns-based, but it also enables us to divide this pruned space into subspaces based on the number of base clusterings used to define the patterns, and search for a consensus solution in each subspace. The proposed approach is explained in detail in the following subsections.

3.1. Clusterings Ensemble

The first step in any consensus clustering method is to build an ensemble of different partitions for the dataset.

However, some consensus methods may impose restrictions on the ensemble generation process. For example, voting methods require that all the partitions in the ensemble have the same number of clusters [8, 21]. For our approach, there are no limitations, as long as the base clusterings define hard partitions, that is, an instance belongs to only one cluster in each base clustering.

3.2. Cluster Membership Matrix

As in [1], we use a cluster membership matrix \mathcal{M} to record the relationships between the instances and the base clusters as a binary relation. \mathcal{M} consists of n rows and m columns, where n is the number of instances, and m is the total number of clusters of all base clusterings.

Definition 1 A cluster membership matrix \mathcal{M} is a triplet $(\mathcal{I}, \mathcal{C}, \mathcal{R})$ where \mathcal{I} is a finite set of instances represented as rows, \mathcal{C} is a finite set of clusters represented as columns, and \mathcal{R} is a binary relation defining relationships between rows and columns: $\mathcal{R} \subseteq \mathcal{I} \times \mathcal{C}$. Every couple $(i, c) \in \mathcal{R}$, where $i \in \mathcal{I}$ and $c \in \mathcal{C}$, means that instance i belongs to cluster c . This binary relation is represented in the matrix by 1 at $\mathcal{M}_{i,c}$, and 0 if there is no relationship.

Consider as an example a dataset of eight instances $\mathcal{D} = \{1, 2, 3, 4, 5, 6, 7, 8\}$. Suppose that we partitioned it with 4 clusterings as follows: $P1 = \{\{1, 2, 3\}, \{4, 5, 6, 7, 8\}\}$, $P2 = \{\{1, 2\}, \{3, 4, 5\}, \{6, 7, 8\}\}$, $P3 = \{\{5, 6, 7, 8\}, \{1, 2, 3, 4\}\}$, and $P4 = \{\{1, 2, 3\}, \{4, 5\}, \{6, 7, 8\}\}$. The resulting cluster membership matrix is shown in table 1. Each column P_j^i defines cluster j in partition i as a binary vector where values ‘1’ identify the instances that belong to the cluster.

Table 1: Example cluster membership matrix.

Instance ID	P_1^1	P_2^1	P_1^2	P_2^2	P_3^2	P_1^3	P_2^3	P_1^4	P_2^4	P_3^4
1	1	0	1	0	0	0	1	1	0	0
2	1	0	1	0	0	0	1	1	0	0
3	1	0	0	1	0	0	1	1	0	0
4	0	1	0	1	0	0	1	0	1	0
5	0	1	0	1	0	1	0	0	1	0
6	0	1	0	0	1	1	0	0	0	1
7	0	1	0	0	1	1	0	0	0	1
8	0	1	0	0	1	1	0	0	0	1

3.3. Frequent Closed Clustering Patterns

\mathcal{M} can be viewed as a pattern mining problem, where each column (cluster) represent an item as defined below.

Definition 2 An item of a cluster membership matrix $\mathcal{M} = (\mathcal{I}, \mathcal{C}, \mathcal{R})$ is a cluster identifier $c \in \mathcal{C}$, and an itemset is a non-empty finite set of items $C = \{c_1, \dots, c_k\} \subseteq \mathcal{C}$ in \mathcal{M} .

The frequency of C is defined as $\mathcal{F}(C) = |\{I \in \mathcal{I} \mid \forall i \in I, \forall c \in C, \text{ we have } (i, c) \in \mathcal{R}\}|$.

Applying FCI technique on \mathcal{M} produces *Frequent Closed Pattern (FCP)*s, that is, a set of instance identifiers that share a binary pattern of cluster memberships (itemset)¹. Table 2 shows the FCPs extracted from table 1². The closed patterns represent maximal rectangles of ‘1’s in the membership matrix.

Definition 3 A pattern $\rho = (C, I)$ in the cluster membership matrix $\mathcal{M} = (\mathcal{I}, \mathcal{C}, \mathcal{R})$ is a pair of sets $C \subset \mathcal{C}$ and $I \subset \mathcal{I}$ such that $\forall i \in I$ and $\forall c \in C$, we have $(i, c) \in \mathcal{R}$. If $C \subset C'$, then ρ is a frequent closed pattern iff $\mathcal{F}(C) \neq \mathcal{F}(C')$.

The closed property ensures not generating redundant patterns for our approach, that is, those with identical instance sets. This reduces greatly memory consumption and execution time compared to generating all the possible frequent patterns³.

Table 2: Frequent closed patterns extracted from table 1.

FCP ID	Itemset (FCIs)	Instance ID set
1	$\{P_2^1, P_2^2, P_2^3, P_2^4\}$	$\{4\}$
2	$\{P_2^1, P_2^2, P_1^3, P_2^4\}$	$\{5\}$
3	$\{P_1^1, P_2^2, P_2^3, P_1^4\}$	$\{3\}$
4	$\{P_2^2, P_2^3\}$	$\{3,4\}$
5	$\{P_2^1, P_2^2, P_2^4\}$	$\{4,5\}$
6	$\{P_1^1, P_1^2, P_2^3, P_1^4\}$	$\{1,2\}$
7	$\{P_2^2\}$	$\{3,4,5\}$
8	$\{P_1^1, P_2^3, P_1^4\}$	$\{1,2,3\}$
9	$\{P_2^1, P_2^3, P_1^3, P_3^4\}$	$\{6,7,8\}$
10	$\{P_2^3\}$	$\{1,2,3,4\}$
11	$\{P_2^1, P_1^3\}$	$\{5,6,7,8\}$
12	$\{P_2^1\}$	$\{4,5,6,7,8\}$

3.4. Multiple Consensuses

After generating the FCPs that summarize the relation between the instances and sets of clustering decisions, we will work on this patterns space to find consensus solutions. First, we partition this patterns space into subspaces based on the number of base clusterings that define the pattern. We use a *Decision Threshold (DT)* to identify each subspace, as it defines the size of the itemset in each pattern.

¹In association rule mining, the *support* of an itemset is the percentage of instances that have the itemset. Only the itemsets with *support* > *minsupport* threshold are considered. However, in our approach, we use *minsupport* = 0 to consider all the possible closed patterns.

²For small datasets, we can have more patterns than the number of instances. However, in large datasets, the number of patterns will be much smaller compared to dataset size, since many instances will share the same pattern.

³See [2] for more details about association rules mining techniques.

The first subspace consists of the patterns having itemsets of size max DT (the number of base clusterings used in the ensemble). The instance identifier sets of the patterns in this initial subspace represent the clusters (“data fragments”) of the first consensus. Next, we sequentially decrement DT toward 1, and in each subspace, a consensus solution is built from the instance identifier sets of the patterns in that subspace, plus the clusters of the previous consensus.

Definition 4 Let $\alpha = \text{Max}(DT)$. The first consensus is $\mathbb{P}^\alpha = \{\pi_1^\alpha, \pi_2^\alpha, \dots, \pi_m^\alpha\}$, where π_k^α is an instance set of a FCP built from α base clusterings. Let $\beta < \alpha$ and $\mathbb{S}^\beta = \mathbb{I}^\beta \cup \mathbb{P}^{\beta+1}$ is the pool of instance sets at $\beta = DT$, where \mathbb{I}^β is the instance sets of the FCPs built from β base clusterings, and $\mathbb{P}^{\beta+1}$ is the instance sets (clusters) of the previous consensus. A new consensus \mathbb{P}^β is the result of applying a consensus function \mathcal{Y} on \mathbb{S}^β , that is, $\mathbb{P}^\beta = \mathcal{Y}(\mathbb{S}^\beta) = \{\pi_1^\beta, \pi_2^\beta, \dots, \pi_n^\beta\}$ such that $\pi_i^\beta \cap \pi_j^\beta = \emptyset$, $\forall (i, j) \in \{1, \dots, n\}$, $i \neq j$, and $\bigcup_{i=1}^n \pi_i^\beta = \mathcal{I}$.

In each subspace, an instance identifier set $I \subseteq \mathcal{I}$ has one of the three following properties:

- i) Uniqueness: It does not intersect with any other set $I' \subseteq \mathcal{I}$, that is, $I \cap I' = \emptyset$.
- ii) Inclusion: It is a subset of another set $I' \subseteq \mathcal{I}$, that is, $I \subseteq I'$.
- iii) Intersection: It intersects with another set $I' \subseteq \mathcal{I}$, that is, $I \cap I' \neq \emptyset$, $I \setminus I' \neq \emptyset$ and $I' \setminus I \neq \emptyset$.

The objective of our consensus function is to build disjoint clusters from the instance sets, that is, all the sets have uniqueness property. For the first consensus, all the instance sets are unique. However, for the following consensuses, the sets of instance identifiers can have any of the above properties, because when we consider fewer base clusterings, instances can belong to several patterns. The instance sets with inclusion property are usually the clusters of the previous consensus, as they will become subsets of new grouping of the instances defined by fewer number of base clusters. Thus, they are removed to consider the new decisions. What remains are the sets with intersection property. To make unique clusters from them, we need to either merge or split intersecting sets based on their similarity. Jaccard index [11] is a well known measure of the similarity between two sets X and Y:

$$J(X, Y) = \frac{|X \cap Y|}{|X \cup Y|} = \frac{|X \cap Y|}{|X| + |Y| - |X \cap Y|}$$

For example, let us take the 3 cases of sets intersection shown in Fig. 1 and calculate the Jaccard score for each case:

$$J(A, B) = \frac{3}{27}, J(B, C) = \frac{7}{23}, J(D, E) = \frac{7}{23}$$

But the same score is given to cases 2 and 3, despite that in case 2, most of set B is part of set C. Thus, instead

of using Jaccard, we define a new measure for deciding to merge or split sets based on the size of intersection to the size of each set:

$$Avg_I(X|Y) = \left(\frac{|X \cap Y|}{|X|} + \frac{|X \cap Y|}{|Y|} \right) \times 0.5$$

Going back to the sets in Fig. 1:

$$Avg_I(A|B) = \left(\frac{3}{20} + \frac{3}{10} \right) \times 0.5 = 0.225$$

$$Avg_I(C|B) = \left(\frac{7}{20} + \frac{7}{10} \right) \times 0.5 = 0.525$$

$$Avg_I(D|E) = \left(\frac{7}{16} + \frac{7}{14} \right) \times 0.5 = 0.469$$

Thus, our new measure gave the highest score to case 2.

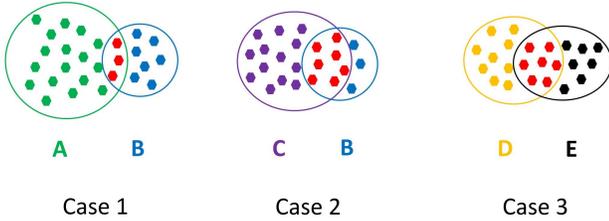


Figure 1: Examples of sets intersection

Based on the new measure, our consensus function will decide to merge/split intersecting sets using a *Merging Threshold (MT)*. That is, if the score given to 2 intersecting sets is more than or equals MT, then the 2 sets are merged using a union operation. Else, they are split into 2 disjoint sets, by removing the shared instances from the largest set, and keeping them in the smaller set as it represent a more coherent cluster. To enhance the merge/split process, the consensus function searches, for each set, which of the other sets produces the highest score by the average intersection ratio measure, before comparing with MT. This process repeats until having all the remaining sets as unique. After generating all the possible consensus, the one the most similar to the ensemble is recommended. The similarity is calculated using Jaccard index. Algorithms 1 and 2 present in detail the proposed multiple consensus generation method.

Continuing the running example and the FCPs in table 2: We start with DT=4. The first consensus consists of the instance sets of FCPs 1, 2, 3, 6, and 9 as they are all unique. The next subspace is DT=3, which consists of FCPs 5 and 8, plus the clusters of the consensus at DT=4. Therefore, we have the following sets: {4}, {5}, {3}, {1,2}, {6,7,8}, {4,5}, and {1,2,3}. The first 4 sets are subsets of the last 2, thus they are removed. The consensus clusters at DT=3 are the remaining sets. At DT=2, we have: {6,7,8}, {4,5}, {1,2,3}, {3,4}, and {5,6,7,8}. The first set is removed, while the second intersects with 2 other sets, thus, we will use average intersection ratio:

$$Avg_I(\{4, 5\}, \{3, 4\}) = 0.5.$$

$$Avg_I(\{4, 5\}, \{5, 6, 7, 8\}) = 0.375.$$

The higher score is for $Avg_I(\{4, 5\}, \{3, 4\})$. If

Input : Dataset to cluster, merging threshold *MT*

Output : ConsTree tree of consensus, list of consensus clustering vectors

- 1 Generate clusterings ensemble of the dataset;
- 2 Build the cluster membership matrix \mathcal{M} ;
- 3 Generate FCPs from \mathcal{M} for *minsupport* = 0;
- 4 Sort the FCPs in ascending order according to the size of the instance sets;
- 5 $MaxDT \leftarrow$ Number of base clusterings;
- 6 $BiClust \leftarrow$ {instance sets of FCPs built from $MaxDT$ base clusters};
- 7 Assign a label to each set in $BiClust$ to build the first consensus vector and store it in a list of vectors $ConsVctrs$;
- 8 **for** $DT = (MaxDT - 1)$ **to** 1 **do**
- 9 $BiClust \leftarrow BiClust \cup$ {instance sets of FCPs built from DT base clusters};
- 10 $N \leftarrow |BiClust|$;
- 11 Call the consensus function (Algo. 2);
- 12 Assign a label to each set in $BiClust$ to build a consensus vector and add it to $ConsVctrs$;
- 13 **end**
- 14 Find stable consensus in $ConsVctrs$ and remove extra duplicates;
- 15 For each remaining consensus, calculate its average similarity to the ensemble using Jaccard index;
- 16 Build a tree from the consensus in $ConsVctrs$, with a recommended solution as the one that has the highest average similarity to the ensemble;

Algorithm 1: Generate Multiple Consensus

$MT=0.4^4$, then the 2 sets are merged to form {3,4,5}. Now we have $Avg_I(\{1, 2, 3\}, \{3, 4, 5\}) = 0.33 < MT$, then the sets are split into {1,2,3} and {4,5}. The last is $Avg_I(\{4, 5\}, \{5, 6, 7, 8\}) = 0.375 < MT$, then split into {4,5} and {6,7,8}. The final consensus at DT=2 is {1,2,3}, {4,5}, and {6,7,8} which is identical to the consensus at DT=3, thus it is removed because it becomes redundant solution. A *stability counter (ST)* is used to reflect that a consensus solution is generated multiple times from different DT spaces. Therefore, the consensus at DT=3 will have $ST=2$. The same process is performed for DT=1, resulting in grouping all the instances in 1 cluster.

3.5. ConsTree

The final step is presenting all the generated consensus in a tree structure that explains how the instances regroup at each DT subspace. Each level in the ConsTree depicts the final consensus clusters of a specific DT subspace. The levels are sorted according to DT, where the first consensus is assigned to the bottom level of the tree. In each tree level, the node's label and size reflect the cluster size. The ConsTree of the running example is shown in Fig.2.

⁴Default value based on extensive experimental tests.

```

1 repeat
2   for i = 1 to N do
3     Bi ← ith set in BiClust;
4     BestIntrsc ← 0;
5     Index ← 0;
6     for j = 1 to N, j ≠ i do
7       Bj ← jth set in BiClust;
8       IntrscSz ← |Bi ∩ Bj|;
9       if IntrscSz = 0 then
10        Next j;
11      else if IntrscSz = |Bi| then
12        /* Bi ⊂ Bj */;
13        Remove Bi from BiClust;
14        Next i;
15      else if IntrscSz = |Bj| then
16        /* Bj ⊂ Bi */;
17        Remove Bj from BiClust;
18        Next j;
19      else
20        IntrscRatio ←
21          (  $\frac{IntrscSz}{|B_i|} + \frac{IntrscSz}{|B_j|}$  ) × 0.5;
22        if IntrscRatio > BestIntrsc then
23          BestIntrsc ← IntrscRatio;
24          Index ← j;
25      end
26    end
27    if BestIntrsc > 0 then
28      j ← Index;
29      Bj ← jth set in BiClust;
30      if BestIntrsc ≥ MT then
31        /* merge */;
32        Bj ← Bi ∪ Bj;
33        Remove Bi from BiClust;
34      else
35        /* split */;
36        if |Bi| ≤ |Bj| then
37          Bj ← Bj \ Bi;
38        else
39          Bi ← Bi \ Bj;
40        end
41      end
42    end
43  until All sets in BiClust are unique;

```

Algorithm 2: Consensus function

Definition 5 A tree of consensus is an ordered set (\mathcal{P}, \preceq) of consensus $\mathcal{P} = \bigcup_{DT=M_{ax}DT}^{DT=M_{in}DT} \mathbb{P}^{DT}$ ordered in descending order of DT values. Let's denote $\mathbb{P}^\alpha = \{\pi_1^\alpha, \dots, \pi_m^\alpha\}$ and $\mathbb{P}^\beta = \{\pi_1^\beta, \dots, \pi_n^\beta\}$ the consensus generated for α and β DT values respectively. Let's denote π_q^α the q^{th} cluster in \mathbb{P}^α and π_r^β the r^{th} cluster in \mathbb{P}^β , with $1 \leq q \leq m$ and $1 \leq r \leq n$. For $\alpha > \beta$ we have $\mathbb{P}^\alpha \preceq \mathbb{P}^\beta$, that is $\forall \pi_q^\alpha \in \mathbb{P}^\alpha, \exists \pi_r^\beta \in \mathbb{P}^\beta$ such that $\pi_q^\alpha \cap \pi_r^\beta \neq \emptyset$. \mathbb{P}^α is a predecessor of \mathbb{P}^β in the tree of consensus.

Another example of a ConsTree is shown in Fig. 3. By reading the tree from the bottom level to the root, we can see that all the clusters are linked to only 1 cluster at the next

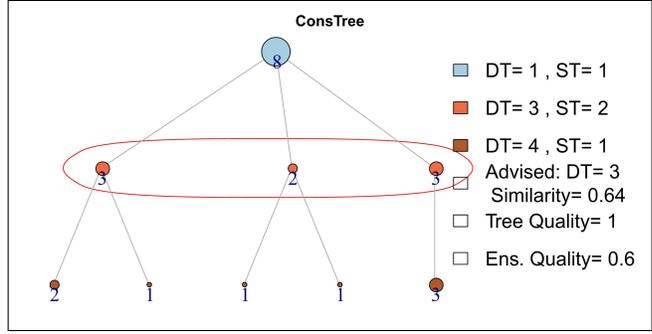


Figure 2: The ConsTree of the running example.

level, except 2 clusters at DT=8 that are linked to 2 clusters at DT=7, meaning that their instances are regrouped differently at DT=7. This shifting of the instances may happen for many of the clusters in other examples (especially for real datasets), making the tree difficult to visualize and analyze. Therefore, we developed a tree refinement process, on which we remove the instances that shift. Figure 4 shows the refined version of the tree in Fig. 3. The *Removed Instances* (RI) at the bottom tells how many instances are removed. Tree refinement do not alter the original consensus, it just simplifies the visualization. What we can understand from the trees is not just how the instances regroup on different view points of a set of base clusterings, but also what clusters (or consensus) are stable. Stable clusters, those that do not change over several consecutive tree levels, suggest strong intra-cluster similarity. Even the merging of the clusters at a higher tree level explains that the merged clusters are very close in the data space compared with others. The recommended solution, the one circled in a red line, is the consensus that has the highest average Jaccard similarity to the ensemble. However, the most stable consensus can also be considered, as it usually identify a well separated clusters structure in the data space. In this example, it is the consensus of DT=5, with stability ST=4.

4. Experiments

The proposed method was implemented using R language [15] on a DELL PRECISION M4800 with Intel® Core™ i7-4710MQ @ 2.50GHz, 32 GB of RAM, and Microsoft Windows 10 Professional (64-bit) operating system. Function *apriori* in *arules* R package [9] was used to discover the FCPs, by setting the *target* parameter to “closed frequent itemsets” and *support* = 1 / Data-size⁵. By considering the clusters of the different consensus as nodes in a graph, that are linked by edges based

⁵A faster algorithm for generating the FCPs called FIST is proposed by [13], and an implementation of it in Java is available on the website of the authors.

The Magic Gamma dataset test is not about the quality of the results, but more about the applicability of our method on large datasets. Note that CLUE methods GV3 and soft/symdiff failed to work on this dataset as they required more than 32 GB of memory. For Magic Gamma, Breast Cancer and EngyTime tests, we considered that we have domain knowledge on how many clusters there should be in the dataset (like differentiating between positive and negative instances). Thus, all the base clusterings generate the same number of clusters, which is the best scenario for voting-based consensus methods.

In table 4, we present the execution time (in seconds) of the consensus methods used. For our method, we separated between the time required to discover the FCPs, and the time required to generate all the consensus and find the recommended one. We can see that the total time of our method is acceptable, although not the fastest compared to CLUE, but much faster than GV3, SM, and Soft/symdiff. In fact, the execution time of our method is related to the size of the ensemble, and the in-ensemble similarity, as both will determine the number of generated FCPs.

5. Conclusions

We presented a new consensus clustering method, using the frequent closed pattern mining technique in order to transform the relation between the instances and the partition ensemble into clustering patterns. By dividing this pattern space into subspaces, we were able to generate multiple consensus from different combinations of base clusterings. In each subspace, we tried to re-cluster the instances based on the information provided in the patterns, rather than finding a “median” partition for the ensemble. The consecutive processing of the different clustering views enables us to discover the number of hidden clusters in the dataset (without the need to specify this explicitly), and to build a ConsTree to visualize the relationships between the instances.

The proposed method achieved good results in terms of quality and the number of discovered clusters, with acceptable execution time considering that it generates multiple solutions. As the FCI technique was designed to efficiently discover patterns in very large datasets, our method can be applied on large datasets, as the patterns space will be a high pruning for the actual instances space. For example, for the Magic Gamma dataset (19020 instances), the patterns space contains 156 patterns only. The number of discovered patterns depends on the size of the ensemble, and how much agreement exists between the base clustering decisions.

Using the ConsTree, the analysts are not limited to one final solution, but rather they can choose another one based on their observation and preferences. For example, they may prefer to choose a solution where a certain cluster is di-

vided into two, as this may reflect a more meaningful grouping for them.

References

- [1] Sitaram Asur, Duygu Ucar, and Srinivasan Parthasarathy. An ensemble framework for clustering protein–protein interaction networks. *Bioinformatics*, 23(13):i29–i40, 2007.
- [2] Aaron Ceglar and John F. Roddick. Association mining. *ACM Computing Surveys*, 38(2), 2006.
- [3] Gabor Csardi and Tamas Nepusz. The igraph software package for complex network research. *InterJournal, Complex Systems*:1695, 2006.
- [4] Sandrine Dudoit and Jane Fridlyand. Bagging to improve the accuracy of a clustering procedure. *Bioinformatics*, 19(9):1090–1099, 2003.
- [5] Xiaoli Zhang Fern and Carla E Brodley. Solving cluster ensemble problems by bipartite graph partitioning. In *Proceedings of the twenty-first international conference on Machine learning*, page 36. ACM, 2004.
- [6] Bernd Fischer and Joachim M Buhmann. Bagging for path-based clustering. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 25(11):1411–1415, 2003.
- [7] Ana LN Fred and Anil K Jain. Data clustering using evidence accumulation. In *Pattern Recognition, 2002. Proceedings. 16th International Conference on*, volume 4, pages 276–280. IEEE, 2002.
- [8] Reza Ghaemi, Md Nasir Sulaiman, Hamidah Ibrahim, and Norwati Mustapha. A survey: Clustering ensembles techniques. *WASET*, 50:636–645, 2009.
- [9] Michael Hahsler, Bettina Gruen, and Kurt Hornik. arules – A computational environment for mining association rules and frequent item sets. *Journal of Statistical Software*, 14(15):1–25, 2005.
- [10] Kurt Hornik. A CLUE for CLUster Ensembles. *Journal of Statistical Software*, 14(12), 2005.
- [11] Paul Jaccard. The distribution of the flora in the alpine zone.1. *New Phytologist*, 11(2):37–50, 1912.
- [12] M. Lichman. UCI machine learning repository, 2013.
- [13] Kartick Chandra Mondal, Nicolas Pasquier, Anirban Mukhopadhyay, Ujjwal Maulik, and Sanghamitra Bandhopadhyay. A new approach for association rule mining and bi-clustering using formal concept analysis. In *Machine Learning and Data Mining in Pattern Recognition*, pages 86–101. Springer, 2012.

Table 3: Tests validation.

Dataset	Magic Gamma	Zoo	E.Coli	Iris	Breast Cancer	Congress Votes	Wine	Wingnut	Terta	EngyTime
Dataset Size	19020	101	336	150	699	435	178	1016	400	4096
# of attributes	10	16	7	4	9	16	13	2	3	2
# of true classes	2	7	8	3	2	2	3	2	4	2
Ensemble size	6	10	9	14	8	8	9	10	9	8
K range	[2]	[3,12]	[4,12]	[2,6]	[2]	[3,6]	[2,6]	[2,6]	[2,7]	[2]
In-ensemble similarity	0.70	0.52	0.53	0.54	0.81	0.66	0.5	0.55	0.59	0.82
Ensemble Min.	0.36	0.27	0.27	0.29	0.36	0.63	0.43	0.40	0.29	0.46
Ensemble Max.	0.46	0.94	0.65	0.88	0.89	0.60	0.89	0.99	1.00	0.88
Our method	0.44	0.82	0.67	0.60	0.87	0.60	0.82	0.94	1.00	0.84
# of clusters in our method	2	9	5	3	2	2	3	2	4	2
SE	0.42	0.76	0.44	0.64	0.86	0.61	0.47	0.88	1.00	0.84
GV1	0.42	0.78	0.44	0.65	0.86	0.61	0.59	0.88	1.00	0.84
DWH	0.42	0.76	0.51	0.59	0.86	0.61	0.53	0.93	1.00	0.83
HE	0.42	0.74	0.47	0.64	0.85	0.60	0.87	0.88	1.00	0.83
GV3	NA	0.76	0.46	0.59	0.86	0.61	0.89	0.92	1.00	0.84
SM	0.41	0.82	0.48	0.63	0.85	0.60	0.50	0.88	0.93	0.83
soft/symdiff	NA	0.83	0.38	0.63	0.85	0.63	0.67	0.92	1.00	0.83
Medoids	0.46	0.84	0.49	0.55	0.86	0.47	0.54	0.99	0.89	0.83

Table 4: Execution time of the consensus methods (in seconds).

Dataset	Magic Gamma	Zoo	E.Coli	Iris	Breast Cancer	Congress Votes	Wine	Wingnut	Terta	EngyTime
Patterns	1.088	0.086	0.196	0.445	0.079	0.141	0.128	0.255	0.117	0.398
Our method	1.299	0.303	2.529	0.751	0.070	1.171	0.534	1.616	0.638	0.555
SE	0.084	0.012	0.027	0.013	0.013	0.011	0.008	0.017	0.016	0.037
GV1	0.113	0.067	0.484	0.050	0.017	0.013	0.103	0.025	0.052	0.035
DWH	0.064	0.007	0.008	0.008	0.008	0.007	0.005	0.012	0.007	0.020
HE	0.144	0.010	0.018	0.013	0.010	0.008	0.010	0.024	0.011	0.030
GV3	NA	0.715	6.306	0.816	13.274	4.276	0.953	25.663	6.388	651.835
SM	33.681	0.783	5.894	0.758	1.399	1.033	0.809	3.296	1.926	10.884
soft/symdiff	NA	5.479	39.051	10.921	138.933	57.454	8.247	377.998	49.040	5639.546
Medoids	0.292	0.039	0.037	0.76	0.036	0.026	0.026	0.127	0.036	0.121

[14] Nicolas Pasquier, Yves Bastide, Rafik Taouil, and Lotfi Lakhal. Efficient mining of association rules using closed itemset lattices. *Inf. Systems*, 24(1):25–46, 1999.

[15] R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2015.

[16] S Sarumathi, N Shanthi, and M Sharmila. A comparative analysis of different categorical data clustering ensemble methods in data mining. *IJCA*, 81(4):46–55, 2013.

[17] Alexander Strehl and Joydeep Ghosh. Cluster ensembles – a knowledge reuse framework for combining multiple partitions. *JMLR*, 3:583–617, 2003.

[18] Alexander P Topchy, Anil K Jain, and William F Punch. A mixture model for clustering ensembles. In *SDM*, pages 379–390. SIAM, 2004.

[19] Alfred Ultsch. Clustering with SOM: U*C. In *Proc. WSOM Workshop*, pages 75–82, 2005.

[20] Sandro Vega-Pons and Paolo Avesani. On pruning the search space for clustering ensemble problems. *Neurocomputing*, 150:481–489, 2015.

[21] Sandro Vega-Pons and José Ruiz-Shulcloper. A survey of clustering ensemble algorithms. *IJPRAI*, 25(03):337–372, 2011.

[22] Ou Wu, Weiming Hu, Stephen J Maybank, Mingliang Zhu, and Bing Li. Efficient clustering aggregation based on data fragments. *IEEE Trans Syst Man Cybern B Cybern.*, 42(3):913–926, 2012.

Mining Feature-Opinion from Reviews Based on Dependency Parsing

Rui Hao, Tieke He, Hang Qi, Jia Liu*

State Key Laboratory for Novel Software Technology, Nanjing University, Nanjing, China

*liujia@nju.edu.cn

Abstract

Manually reading all the product reviews to find a satisfying item is not only labor-intensive, but also tedious for the consumers. In this paper, we propose a feature-opinion mining approach to automatically summarize the reviews. Specifically, in our approach we first utilize a regression model to generate sentiment word, including phrase and its sentiment weight, then extract feature based on the dependency relationship between feature word and sentiment word, and finally we assign score to feature according to the dependency relationship. The experimental results demonstrate that our approach can effectively mine the feature-opinion from reviews.

Keywords: E-commerce, opinion analysis, dependency parsing

1. Introduction

In the information age, electronic commerce has been widely accepted by the public. According to the Economic Daily¹, there have been more than 0.3 billion netizens and 83 million online markets contributing to electronic commerce in China.

However, there is a serious problem in e-commerce, it is the intermingling of good, mediocre and bad products. Consumers usually judge whether a product is good or not by its online reviews, but too many reviews bring them trouble in choosing a suitable product within 10 minutes [1, 7], even though they clearly know what they want.

Facing enormous comments, people are desperate for a tool to scan and summary the comments automatically. Researchers have proposed many methods [2, 6, 9] for extracting product virtues and defects. In this paper, we propose feature-opinion mining approach, which is based on dependency parsing. In summary, there are two main contributions of our work:

¹http://paper.ce.cn/jjrb/html/2014-05/31/content_202681.htm

Feature extraction: Previous works usually identify features according to the lexical categories of the word, for example, M. Hu [6] used the CBA association rule miner [8] to pick out nouns or noun phrases in the review as possible features, but some non-noun features would be omitted in their method. In this paper we extract feature in a new way that we regard the word or phrase which is described or modified most frequently by sentiment words as feature candidates. The main idea behind this approach is that when people talk about feature, they usually use some opinion words on the feature, in other words, if a word or phrase is related to many opinion words, it is probably a feature.

Feature scoring: In the work of C. Scaffidi, et al [9], they assigned the semantic orientation of the whole sentence to features in the sentences as their orientation, which is not suitable for our work, as the reviewer may give both terrible and fine features in a single sentence. In that case, terrible features should be assigned negative orientation while fine features should be assigned positive orientation. Therefore in this article we compute semantic score for each feature in a sentence. What's more, we compute the score using the orientation sum of all opinion words which have relationship with the feature. Different from previous works [3], which defined the relationship as text distance between feature and opinion word, our work prefer to use syntax dependency between feature and opinion word, which has been proved more accurate [10].

The rest of this paper is organized as follows. We first present our approach in detail in next section, then we evaluate our method in Section 3, and finally we conclude this paper and outline some future work in Section 4.

2. Approach

2.1 Sentiment Word Generation

We collected product information and consumer reviews in December 2015 from JingDong website², which is one of

²<http://www.jd.com>

the largest Chinese online shopping platforms. We crawled 1,909 cellphone products, 1,089 digital single lens reflex camera(DSLR) products and 3,160 tablet products, whose review count is 357,126, 53,980 and 308,574 respectively.

We utilized a linear regression model [5] to generate sentiment words, it models the relationship between review text and rating, and output weights for each word, which is crucial for us to calculate the score of features. We got the items with outstanding positive and negative weights as our sentiment words.

Instead of dumping data into model directly, we first segmented the review text using *jieba* tool³, and then we removed stop words, transformed the English words to lowercase, and finally filtered out the words whose frequency of occurrence is less than 1,000. After the pre-processing steps, a corpus, which would later be formalized into a $m \times n$ matrix X , was obtained. Here, m is the number of comments, and n is the size of corpus. We also need a vector Y with m items to stand the ratings. Then we trained the following regression model, X_{ij} means the frequency of the j -th word in the i -th comment, Y_i is the rating of the i -th comment.

$$L(W) = \sum_{i=1}^m (Y_i - (w_0 + \sum_{j=1}^n X_{ij}w_j))^2 + \gamma P(W) \quad (1)$$

$$P(W) = \sum_{i=1}^n ((1 - \alpha)w_i^2 + \alpha|w_i|) \quad (2)$$

Through the model above we could produce a set of parameters $W = w_0, w_1, \dots, w_n$ as the words' weight.

We used *glmnet* package in R [4] to do the linear analysis, and the best γ for category cellphone, DSLR, tablet is 0.00041, 0.00390 and 0.00061 respectively. The α did not need to be optimized and was set to 1.

Finally, we listed the words with the largest positive weights and negative weights to constitute our sentiment word set. The top 5 sentiment words for each category are listed in Figure 1.

2.2 Feature Extraction

We have obtained the sentiment words for each category in above-mentioned steps, and we will present how to extract features using those sentiment words in this section.

2.2.1 Dependency parsing

Before describing the feature extraction method, we first need to introduce the dependency parsing, which reveals the syntactic structure by analyzing the dependency relationship between different components of a sentence. That

³<https://github.com/fxsjy/jieba>

is, the dependency parsing labels grammatical constituents such as “subject-verb”, “verb-object” and analyzes the relations between them.

To do the parsing job, we utilized analysis service provided by HIT-SCIR⁴ in this article. A simplified example is illustrated in Figure 2. From the figure we can see that the head predicate of the sentence is “great”, the subject is “phone”, “so” is degree adverbial of “great” and “this” modifies “phone”. The whole list of annotation of dependency relationships used in this article is listed in the Table 1.

Table 1. Dependency relationship definition

Tag	Description	Example
HED	head	the core part of the sentence
SBV	subject-verb	I gave her a camera (I ← gave)
VOB	verb-object	I gave her a camera (gave → camera)
ATT	attribute	great camera (great ← camera)
ADV	adverbial	so great (so ← great)

2.2.2 Extracting feature

For each sentence in a review, we captured its syntactic structure based on dependency parsing, which has been mentioned above, and saved the parse result to a .xml file. Then we defined rules to extract feature candidates from the result files.

In this paper, we define five extraction rules that are listed in Table 2. s and f refers to sentiment word and feature candidate respectively, tmp means any word, S (or F) represents sentiment words (or feature candidates) which have been extracted before and ref stands for the dependency relationships.

After feature extraction, we applied feature combination to make the final feature set more accurate. We merged two features according to their similarity, whose calculating formula is listed in the following. f_i, f_j stands for different features, and $union(f_i, f_j)$ means the length of common parts in f_i and f_j , $max(f_i, f_j)$ means the max of f_i 's length and f_j 's length. For feature whose length was 2, 3, 4 and 5, we chose 1.0, 0.6, 0.5 and 0.6 as their threshold respectively.

$$sim(f_i, f_j) = \frac{union(f_i, f_j)}{max(f_i, f_j)} \quad (3)$$

Based on the frequency of occurrences, we list the top 20 hottest features of cellphone, DSLR and tablet in Figure 3. In particular, we take a descriptive word for each feature, which is written in Chinese, to express the main idea of it.

2.3 Feature Scoring

For a feature, we summed weights of all sentiment words that had dependency relations with it as its score. Assume

⁴<http://www.ltp-cloud.com>

	Cellphone				DSLR				Tablet			
	word	description	weight	frequency	word	description	weight	frequency	word	description	weight	frequency
Positive	差不多	fine	0.543	1132	不错	nice	0.168	21504	差不多	almost	0.502	1351
	国货	domestic	0.291	2002	行货	authentic	0.109	1405	不卡	smooth	0.482	2853
	放心	assured	0.217	1085	实用	practical	0.088	1440	正品	certified	0.276	11058
	玩游戏	games	0.201	1085	正品	certified	0.071	8915	价钱	price	0.27	4317
	不错	good	0.191	111010	耐心	patient	0.07	1010	一天	a day	0.267	2079
Negative	失望	disappointed	-0.505	1458	不	no	-0.14	3622	后悔	regret	-0.583	1360
	坏	broken	-0.453	3061	客服	custom-service	-0.139	2871	太差	poor	-0.53	1081
	垃圾	rubbish	-0.427	3455	货	product	-0.097	1017	垃圾	rubbish	-0.502	4561
	死机	crash	-0.419	1796	发票	receipt	-0.094	1143	不好	bad	-0.5	6353
	经常	frequent	-0.344	1342	京东	JingDong	-0.091	3488	一个月	a month	-0.495	1368

Figure 1. Top 5 sentiment words of cellphone, DSLR and tablet

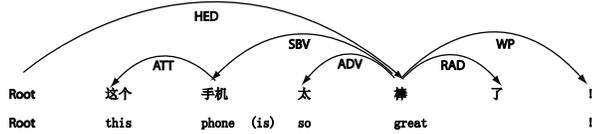


Figure 2. Dependency parsing example

that there is a review named “great cellphone”, from the dependency result we know that word “great” modifies word “cellphone”, and the orientation weight of “great”, which is calculated in Section 2.1, is supposed to be 0.5, so we say that the score of feature “cellphone” in review “great cellphone” is 0.5.

We obtained feature list for each product category in Section 2.2, then for each feature f on the list, we calculated its score on product p according to the equation:

$$score(f, p) = \frac{1}{n} \sum_{r \in R_p} \sum_{s_i \in S_r^f} weight(s_i) \quad (4)$$

R_p is the set of all reviews of product p , S_r^f is the set of sentiment words of feature f in review r , s_i and $weight(s_i)$ represents the sentiment word and its orientation weight, n means the size of set R_p .

Getting the set S_r^f is very important for calculating $score(f, p)$, and we also defined rules to detect sentiment word in a review, which has been listed in Table 2.

3. Evaluation

ZOL⁵ is a specialized information provider of digital products, it offers lots of information, including expert comments, customer comments, user forums and market news. In particular, ZOL extracts some features as standard features for each category manually and requires reviewers to rate them as long as they want to publish a review on the website. The consumers’ ratings are the capable criterion

⁵<http://www.zol.com.cn>

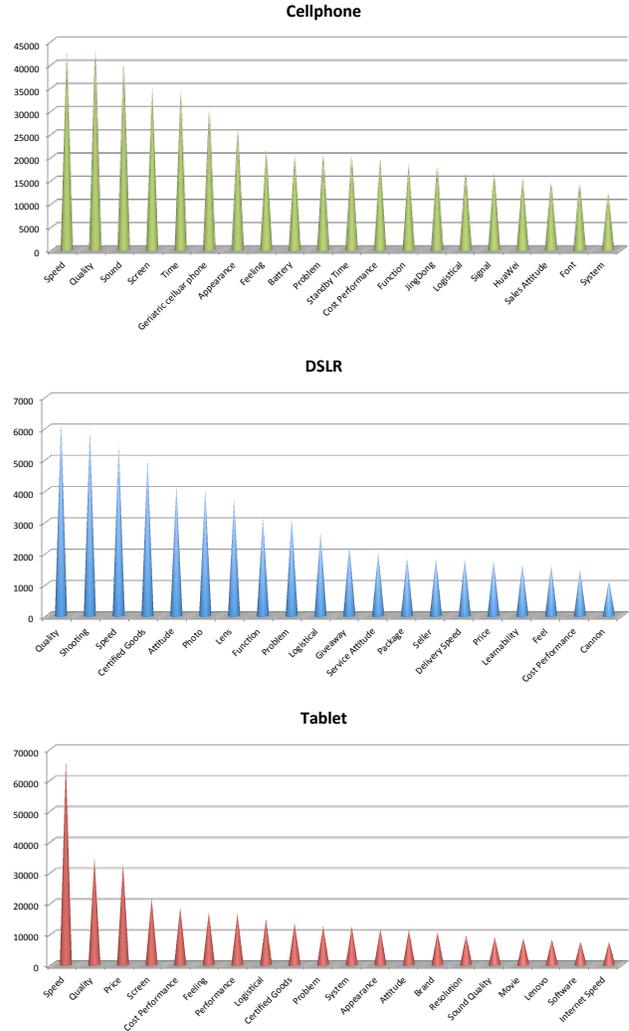


Figure 3. Top 20 hottest features

to evaluate our scoring result, so we crawled ratings for category cellphone, DSLR and tablet from ZOL in December 2015, and then matched product of ZOL with product of JingDong manually. After matching process, we obtained a date set with 120, 75 and 126 products for category cellphone, DSLR and tablet, respectively.

Table 2. Rules for feature candidate extraction

	Pattern	Constraint	Conclusion
R1	$f \rightarrow ref \rightarrow f'$	$ref \in \{ATT\}, f' \in F$	$f \in F$
R2	$f' \rightarrow ref \rightarrow f$	$ref \in \{ATT\}, f' \in F, f \notin S$	$f \in F$
R3	$f \rightarrow ref \rightarrow s$	$ref \in \{ADV, ATT\}, s \in S(f \in F)$	$f \in F(s \in S)$
R4	$s \rightarrow ref \rightarrow f$	$ref \in \{SBV, VOB, ATT\}(ref \in \{VOB, ATT\}), s \in S(f \in F)$	$f \in F(s \in S)$
R5	$f \leftarrow ref \leftarrow tmp \rightarrow ref' \rightarrow s$	$ref \in \{SBV\}, ref' \in \{VOB\}, s \in S(f \in F)$	$f \in F(s \in S)$

Then we matched standard features in ZOL with features we extracted before, but some standard features in ZOL are too general to match, for example there is a standard feature named “Entertainment”, but entertainment contains many aspects, such as movie, game, song, internet, etc. So we filtered out these features and got 5 cellphone features, 4 DSLR features and 5 tablet features as our criterion finally.

We utilized a generalized precision metric[9], which is listed in the following, to measure the difference between system score set S_f^c of category c and feature f with human score set H_f^c , n is the size of S_f^c , which is equal to the size of H_f^c . The evaluation result is illustrated in Figure 4.

$$precision(c, f) = 1 - \frac{1}{n} \sum_{i=1}^n \frac{|s_i - h_i|}{MAX(s_i, h_i)} \quad (5)$$

where $s_i \in S_f^c, h_i \in H_f^c$

4. Conclusion and Future Work

In this paper, we proposed a method to summarize user reviews based on dependency parsing. Our approach performed well during evaluation but there is still room for improvement. In our future work, we plan to add new feature extraction and feature scoring rules to refine the result, and we also intend to improve our feature combination using synonyms or semantic analysis.

References

[1] L. Barnard and J. L. Wesson. Usability issues for e-commerce in south africa: an empirical investigation. In *SAICSIT*, 2003.

[2] K. Dave, S. Lawrence, and D. M. Pennock. Mining the peanut gallery: Opinion extraction and semantic classification of product reviews. In *WWW*, pages 519–528, 2003.

[3] X. Ding, B. Liu, and P. S. Yu. A holistic lexicon-based approach to opinion mining. In *WSDM*, pages 231–240, 2008.

[4] J. Friedman, T. Hastie, and R. Tibshirani. Regularization paths for generalized linear models via coordinate descent. *JSS*, 33(1):1, 2010.

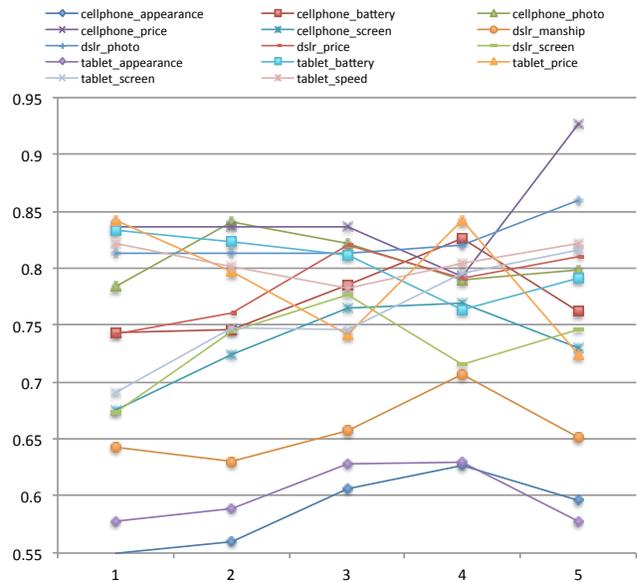


Figure 4. Feature scoring evaluation result

[5] B. Fu, J. Lin, L. Li, C. Faloutsos, J. Hong, and N. Sadeh. Why people hate your app: Making sense of user feedback in a mobile app store. In *KDD*, pages 1276–1284, 2013.

[6] M. Hu and B. Liu. Mining and summarizing customer reviews. In *KDD*, pages 168–177, 2004.

[7] R. Kohavi and R. Parekh. Ten supplementary analyses to improve e-commerce web sites. *Proceedings of the Fifth Webkdd Workshop*, 2003.

[8] B. L. W. H. Y. Ma. Integrating classification and association rule mining. In *KDD*, 1998.

[9] C. Scaffidi, K. Bierhoff, E. Chang, M. Felker, H. Ng, and C. Jin. Red opal: Product-feature scoring from reviews. In *EC*, pages 182–191, 2007.

[10] L. Zhuang, F. Jing, and X. Y. Zhu. Movie review mining and summarization. In *CIKM*, pages 43–50, 2006.

Experimentation in the Industry for Automation of Unit Testing in a Business Intelligence Environment

Igor Peterson Oliveira Santos^a, André Vinícius R. P. Nascimento^b, Juli Kelle Góis Costa^a, Methanias Colaço Júnior^{a,b}, Wenderson Campos Pereira^b

^a Postgraduate Program in Computer Science - PROCC.

UFS – Federal University of Sergipe
São Cristóvão/SE - Brasil.
{igorp.ita, julikelle}@hotmail.com

^b Competitive Intelligence Research and Practice Group – NUPIC

Information Systems Department - DSI
UFS – Federal University of Sergipe
Itabaiana/SE - Brasil.
andreviniciusnascimento@gmail.com, {mjrs,
wenderson_se}@hotmail.com

Abstract— This paper presents an approach to automate the selection and execution of previously identified test cases for loading procedures in Business Intelligence (BI) environments based on Data Warehouse (DW). To verify and validate the approach, a unit test framework was developed. The overall goal is achieve data quality improvement. The specific aim is reduce test effort and, consequently, promote test activities in data warehousing process. A controlled experiment evaluation was carried out to investigate the adequacy of the proposed method for data warehouse procedures development. The results of the experiment show that our approach clearly reduces test effort when compared with manual execution of test cases.

Keywords— *Business Intelligence; Data Warehouse; Software Testing; Data Quality; Experimental Software Engineering.*

I. INTRODUCTION

Information represents a crucial factor for companies in improving processes and decision making. To assist the strategic areas of the organizations business intelligence (BI) environments are presented as sets of technologies that support the analysis of data and key performance indicators [1]. A central component of BI systems is a Data Warehouse (DW), a central repository of historical data. The idea behind this approach is to select, integrate and organize data from the operational systems and external sources, so they can be accessed more efficiently and represent a single view of enterprise data [1, 2, 3].

Despite the potential benefits of a DW, data quality issues prevent users from realizing the benefits of a business intelligence environment. Problems related to data quality can arise in any stage of the ETL (Extract, Transform and Load) process, especially in the loading phase. The main causes that contribute to poor data quality in data warehousing are identified in [4]. The lack of availability of automated unit testing facility in ETL tools is also appointed as cause for the poor data quality [4]. The low adoption of testing activities in DW environment is credited to the differences between the architecture of this environment and architectures of the generic software systems. These differences mean that the testing techniques used by the latter need to be adjusted for a DW environment [5, 6].

The test of ETL procedures is considered the most critical and complex test phase in DW environment because it directly affects data quality [7]. ETL procedures, more precisely the

loading routines, exhibit the same behavior as database applications. They operate on initial database state and generate a final consistent database state. So, a black-box approach, which combines the unit and application behavior of loading procedures, is proposed. In this approach, the concern is with the application interface, and not with the internal behavior and program structure [8,9]. This approach to ETL routines, in some environments, may be the only option, since the use of ETL tools in DW environment produces codes or packages whose internal structure is not known.

This paper presents the result of the construction and experimentation of a unit testing framework to improve the quality of loading procedures in a BI environment. Using a black-box approach and treating the loading routines under the application point of view, the framework generates initial and final states of the database in function of test cases previously specified. The expected behavior of the routines, the selection of test cases and database conditions used are determined by procedures metadata.

II. FRAMEWORK

A. Architecture

The framework will be used to perform tests under the black-box approach. The code and the internal structure of the routines will not be examined. The test cases previously implemented, will be selected according to the characteristics of the routine being tested. Each routine, in order to be covered by the framework, must have a set of metadata registered. This set of metadata was defined from the schema presented in [10].

B. FTUnit tool

The FTUnit tool is a framework used to perform unit tests in loading procedures of a DW environment. It has been developed in C#. More details about the tool and download are available at <<http://ftunit.wordpress.com/>>.

C. Goal definition

Our work is presented here as an experimental process. It follows the guidelines by Wohlin et al. in [11]. In this section, we start introducing the experiment definition and planning. The following sections, will direct to the experiment execution and data analysis.

Our main goal is to evaluate the use of automated test execution to loading routines in a Data Warehouse environment.

The experiment will target developers of ETL processes for BI environments with at least 2 years of experience in the market and one year of experience in ETL programming. The goal was formalized using the GQM model proposed by Basili and Weiss [12]: **analyze** the use of a DW unit testing framework, **with the purpose of** evaluate (against manual testing), **with respect to** the efficiency of the process of executing test cases, **from the point of view of** developers and decision support managers, **in the context of** programmers in a BI company.

D. Planning

1) Hypothesis Formulation

The research question for the experiment that needs to be answered is this: “A customized unit testing framework can increase the productivity of developers during the testing process in a DW?” To evaluate this question, it will be used a measure: Average time for Manual testing and Automation Testing. Having the purpose and measures defined, it will be considered the hypothesis: **1) H_{0time}** - the execution of automated and manual testing has same efficiency. ($\mu_{ManualTestingTime} = \mu_{AutomationTestingTime}$); **2) H_{1time}** - the execution of automated testing is more efficient than the execution of manual testing. ($\mu_{ManualTestingTime} > \mu_{AutomationTestingTime}$).

Formally, the hypothesis we are trying to reject is H_{0time} . To ascertain which of the hypotheses is rejected, will be considered the dependent and independent variables that can be seen as follow.

a) Independent Variables

Next, the independent variables of the experiment are described.

Description of Test Cases Used in the Experiment: The loading routines for the DW environment are quite discussed in [1,2]. Alternative approaches to the loading of dimensions can be found in [13]. Algorithms for loading routines for the various types of dimensions can be found in [14]. Test Cases categories for ETL routines are pointed in [6]. This material, together with the extensive experience of the authors in DW projects in the public and private sectors, provided the basis for the elaboration of categories and test cases to be considered by the framework. The following categories are contemplated by the framework: a) Unit tests and relationship; b) Number of records between source and destination; c) Transformations between source and destination; d) Processing of incorrect or rejected data; e) Null values processing; f) Behavior type 1, type 2 and type 3¹ for dimensions attributes; g) Hybrid approaches for the treatment of historical dimensions.

Description of the Use Case Used in the Experiment: the characteristics of the use case chosen for the validation study were based on practical situations reported by the selected programmers. For the use case of the experiment, the goal was to generate procedure to perform loads of Staging Area, from employee table to the employee dimension. At this time, the dimension has an historical storage, Type 2 for some attributes. The other attributes are Type 1.

¹ There is treatment of historical data, by storing in various attributes of the same data record. So, can be created how many columns as desired for the dimension [13].

Tests in the ETL Process: The ETL tests used in this work, have two types of treatments for performing the experiment: 1) Manual Testing: manual testing execution, based on the test cases, in the ETL procedures, in SQL to load data defined in the use case already presented earlier; 2) Automation Testing: execution of tests based on test case, in the SQL code for the same use case, using the proposed tool in this work, FTUnit.

b) Dependent Variables

It were used a measure as a dependent variable: Average time, for manual testing and automation testing, measured using a stopwatch, considering the average time spent on testing in the procedures.

2) Participants Selection

The selection process of the participants will be done for convenience, making the type of sampling per share in which will be preserved the same characteristics of interest present in the population as a whole. The contributor to be chosen will be Qualycred (www.qualycred.com), a company that provides consulting in BI solutions for industry. This company will provide for the execution of the experiment, ten programmers with four years of experience in other areas and one year of experience working specifically with ETL for DW, in SGBD SQL SERVER.

3) Experiment Project

The experiment was projected in a paired context, in which a group will evaluate both approaches: Manual and Automated execution of test. For understanding the execution of the test to be done, ten test cases and one use case (seen in Independent Variables section) were elaborated, which will be presented in a well detailed way to the programmers.

The experiment will be separated into two groups of participants. Will be drawn 5 programmers to start the tests to the rules presented in the Employee Use Case, with the execution of manual testing and, shortly after, the execution of automation testing. The other participants in parallel, will make the tests made to rules presented in same use case, with the implementation of the automation testing and, shortly after, with the execution of the manual testing. Thus, the randomness will be enhanced, not prioritizing the manual or automated learning.

4) Instrumentation

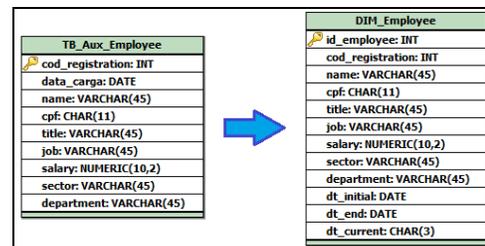


Figure 1. Charge for a dimension with behaviors of types 1 and 2.

The instrumentation process initially proceeded with the environment setup for the experiment and planning the data collect. It was conducted in a computer lab at Federal University of Sergipe - UFS.

Figure 1 contains the representation of a load data from TB_Aux_Employee (auxiliary table of employees) to the DIM_Employee (dimension of employees) that represents a

dimension of types 1 and 2. To this dimension, the attributes that match the type 1 are: name and CPF. The attributes of type 2 are: title, job, salary, sector and department.

III. EXPERIMENT OPERATION

A. The preparation

The following are listed the preparation steps for the execution of the experiment. 1) *DW environment Creation* - in this phase was defined and created the DW environment with the dimensional schemes and staging area; 2) *Definition of Test Cases for loading routines* - were defined test cases to be followed by the developers of the experiment. 3) *Review of basic concepts of loading routines for the programmers* - a review of the loading routines, for DW environments with the selected developers, was performed. 4) *Training in testing framework* - a training with the programmers was realized to become familiar with the tool.

In short, all computers were prepared with the same settings, so programmers were on the same working conditions. Moreover, it was presented to each programmer, a printed document containing a detailed description of Use Case and test cases that would be used by them, in case of any doubts.

B. Execution

At the end of the previous steps, the experiment was initiated, it occurred according to the plan described in section 3. The evaluation of the tool at the end of the experiment, made by the professionals, was positive, since they have commented that the use of the tool have contributed to the reduction of time in the test procedures.

1) Data Collection

It was calculated the time spent by each developer for both manual and automated tests, of all test cases for the Employee Use Case, taking into account the time for testing and all necessary settings in FTUnit. Under supervision, each programmer reported the completion and was recorded the time on a timer, used for this purpose. The result of these collected data will be presented in section 5 of this paper.

C. Data Validation

In order to perform the experiment, one factor was considered, Test of the ETL Process, and two treatments, manual and automated tests, using the FTUnit tool. Facing this context, the average of testing time was computed.

As an aid to analysis, interpretation and validation, we used two types of statistical tests, Shapiro-Wilk Test and the T Test. Shapiro-Wilk test was used to verify normality of the samples. The T test was used to compare the average of the two paired samples [11]. All statistical tests were performed using the SPSS tool [15].

IV. RESULTS

A. Analysis and Data Interpretation

To answer the question of research, the following dependent variable was analyzed: The time to the testing process of each procedure.

1) Time spent in the testing process.

Results - related to the testing time by each participant for the Employee Use Case - show that the average time of the

developers for manual testing was 54.4 minutes, and 20.5 minutes for the automatic one.

These results suggest that the automated testing procedures have, on average, shorter testing time, as compared with the same test procedure performed manually by programmers with experience in the area. Thus, from this preliminary analysis of the data, it is assumed that the answer to the Research Question would be "yes". The execution of automated testing can increase the productivity of developers during the testing process in a DW, since automation testing obtained a difference of approximately 35 minutes. But is not possible to make such a claim without sufficiently conclusive statistical evidence.

Thus, first, we established an apriority significance level of 0.05. The Shapiro-Wilk test ensured that the sample was normally distributed. We found p-values of 0.659 and 0.311 for execution of manual and automated testing, respectively. As the p-value is the lowest possible significance with which it is possible to reject the null hypothesis, and they are larger than 0.05, we cannot reject the hypothesis that the data is normally distributed.

Finally, as the samples are not independent, the hypothesis test applied in this context was the T-Test, characterized as parametric for paired samples, which only requires normality of the samples. We obtained the p-value of 0.000. This means the p-value found is less than 0.0001, so we have more than 99% certainty for the valued context. Thus, it was confirmed the evidence of a difference between the averages of 33.9. As the significance test is lower than 0.05, it is possible to reject the null hypothesis. Consequently, we cannot reject the alternative hypothesis that the execution of automated testing is more efficient than the execution of manual testing.

B. Threats to validity

In spite of having achieved statistical significance in the study, the following threats to the validity must be considered.

Threats to internal validity: Although participants have been trained to use the tool, they do not use it daily. This lack of constant contact with it may have affected the results, which could be even better, pro-tool. The tool training was conducted at the beginning of the experiment, considering a phenomenon studied by psychology called *Demand Characterization* - which considers that an experimental artifact may have an interpretation of the purpose of the experiment by the participants. This can lead to change of unconscious behavior, to adapt to this interpretation [16]. According to this concept, this training could be harmed the progress of the experiment, but to mitigate this factor, can be said that had been used at least two different approaches: *The More The Merrier* and *Unobtrusive Manipulations and Measures* [16]. Respectively, the first, to avoid bias with a single experimenter, the experiment had another researcher to conduct the experiment and an instructor for the tool, not involved with the research. The second guided us not to say which factors and metrics would be assessed, so that the participants had no clues about the research hypothesis.

Threats to external validity: The low number of participants can be a threat, since it can negatively influence the results of the experiment. This threat was mitigated with the

convenience of the selection of programmers skilled in the ETL area for BI environment.

Threats to the construction validity: The Specifications for the use case and test cases may not have been very clear to the understanding of some programmers. This threat was mitigated with the prior reading and analysis of the understanding, made by 3 ETL developers.

V. RELATED WORK

Through literature reviews, with systematic approaches, were not found strongly related work for automated unit tests in ETL tools. Consequently, the absence of ETL tools with these characteristics may contribute to a lower integrity and a lower quality of data, essential in large banks of decision support data.

Some moderately related works also seek solutions for the automatic execution of Test Cases in DW environments. In [6] it is presented a directed models approach for automatic generation and execution of test cases based in formal models of systems. The formal model adopted is based on the UML language. The approach also depends on creating an extension of UML language that can capture the transformations used in a *Data Warehousing* process.

The QuerySurge [17] tool, developed by RTTS Company, presents the possibility of automatic execution of unit tests. This approach differs from ours, since the goal is to work with programmed unit tests, not pre-defined by the tool. The approach adopted is to create scripts that can capture operational environment data and dimensional schema for comparison. The tool does not use metadata to work with already known transformations, as it happens in our approach.

Once the framework generates test cases based on characteristics of the loads procedures being implemented, it can be extended and used to test load routines created for any ETL tools. So far was not found in literature any similar approach, so we could make a comparison. The more similar tool to the proposed work is the framework [18]. However, this one represents a generic framework for database applications and has no particularity regarding to loading routines for a Data Warehouse environment.

VI. CONCLUSIONS

Business Intelligence requires valid, consistent, and complete organizational data. These quality items represent constant concerns for companies in the process of use of decision support systems.

In this paper, we presented the proposal of using a unit testing framework for loading routines in a BI environment based on Data Warehouse. The motivation for adopting this approach meets the problems pointed out in [19], as the main causes for the poor quality of data in a DW environment. Another motivation, also pointed in [6,7,8] is the need to adopt different strategies, considering the differences between traditional environments and DW environments, which can contribute to the adoption of testing processes.

In this context, this work presents important contributions to increasing the productivity and quality in software engineering for loading routines of DWs, and encourages experimentation in an industrial environment. The framework encapsulates a method to accelerate and improve the quality of ETL process

tests based on SQL. It is noteworthy that the safe and efficient execution of procedures in SQL directly in the database is an option considered by much of the industry, requiring tools to support tests in this type of approach in software engineering.

The proposed framework presents test cases previously defined which cover the main categories of tests applied to loading routines. Through a set of metadata that defines the characteristics of the routines, the framework selects test cases to be applied, generates the initial states of the database, executes the routines, performs test cases, analyzes the final state of the database and generates a report with the errors encountered during the execution of each test case.

By virtue of what we have seen above and the framework innovation, the presentation of this experiment will support the adoption of the same or the creation of a similar approach for companies that use this type of strategy.

As future work, experiments will be done evaluating the use of the proposed framework against a generic database application test framework, the DBUnit [18] which had been constructed specifically for database application tests.

REFERENCES

- [1] Colaço Jr., M.: *Projetando sistemas de apoio à decisão baseados em Data Warehouse*. 1st ed., Rio de Janeiro: Axcel Books (2004)
- [2] Kimball, R., Ross, R. M. and Thomthwaite, W.: *The Data Warehouse lifecycle toolkit*. 2nd. ed., Indianapolis, Indiana: Wiley Publishing Inc (2008)
- [3] Inmon, W. H.: *Building the Data Warehouse*. 4th ed., Indianapolis, Indiana: Wiley Publishing Inc (2005)
- [4] Ranjit S. and Kawaljeet, S.: *A Descriptive Classification of Causes of Data Quality Problems in Data Warehousing*. 7 v. IJCSI International Journal Of Computer Science Issues (2010)
- [5] Deshpande, K.: *Model Based Testing of Data Warehouse*. IJCSI International Journal of Computer Science Issues, Vol. 10, Issue 2, No 3 (2013)
- [6] Elgamel, N., Elbastawissy, A. and Galol-edeem, G.: *Data Warehouse Testing*. EDBT/ICDT '13, Genoa, Italy (2013)
- [7] Golfarelli, M. and Rizzi, S. A.: *Comprehensive Approach to Data Warehouse Testing*. ACM 12th International Workshop on Data Warehousing and OLAP (DOLAP '09), Hong Kong, China (2009)
- [8] Myers, G. J., Badgett, T. and Sandler, C.: *The Art Of Software Testing*. 3rd ed., New Jersey: Wiley (2012)
- [9] Sommerville, I.: *Engenharia de Software*. 9th ed., São Paulo: Pearson (2011)
- [10] J. K. G. Costa, I. P. O. Santos, A. V. R. P. Nascimento, M Colaço Jr. *Experimentação na Indústria para Aumento da Efetividade da Construção de Procedimentos ETL em um Ambiente de Business Intelligence*. SBSI 2015, May 26–29, Goiânia, Goiás, Brazil (2015)
- [11] Wohlin, C., et al.: *Experimentation in Software Engineering: An introduction*. USA: Kluwer Academic Publishers (2000)
- [12] Basili, V. and Weiss, D.: *A Methodology for Collecting Valid Software Engineering Data*. In: *IEEE Transactions On Software Engineering*, v.10 (3): 728-738, November (1984)
- [13] Santos, V. and Belo, O.: *No Need to Type Slowly Changing Dimensions*. IADIS International Conference Information Systems (2011)
- [14] I. P. O. Santos, J. K. G. Costa, A. V. R. P. Nascimento, M. Colaço Júnior. *Desevolvimento e Avaliação de uma Ferramenta de Geração Automática de Código para Ambientes de Apoio à Decisão*. In: XII WTICG, XII ERBASE (2012)
- [15] SPSS, IBM Software, <http://goo.gl/eXfcT3>
- [16] Orne, M. T.: *Sobre a psicologia social da experiência psicológica: Com referência particular para exigir características e suas implicações*. (1962)
- [17] QuerySurge, RTTS, <http://www.querysurge.com/>
- [18] DBUnit, <http://dbunit.sourceforge.net/>
- [19] Singh, R. and Singh, K.: *A Descriptive Classification of Causes of Data Quality Problems in Data Warehouse*. IJCSI International Journal of Computer Science Issues, Vol. 7, Issue 3, No 2, May (2010)

Automatically Finding Hidden Industrial Criteria used in Test Selection

Cláudio Magalhães
Centro de Informática
Recife, Brazil
cjasm@cin.ufpe.br

Alexandre Mota
Centro de Informática
Recife, Brazil
acm@cin.ufpe.br

Eliot Maia
Motorola Mobility
Jaguariúna, Brazil
eliotm@motorola.com

Abstract

In this paper we propose a way to find weights of a ranking function semi-automatically. From the manual choices made by (experienced) human test architects, our idea is to propose an optimization model that tries to find the necessary weights automatically. We present some experiments by encoding our optimization model in the Z3 SMT solver and using real Motorola Mobility¹ data.

1 Introduction

Regression testing is vital for any product development. It gives a measure of how the adjustments made preserved accepted functionality. Re-run all available test cases is the easiest alternative for a regression testing campaign. In practice, however, this is usually unfeasible due to the effort needed in terms of time and human (for manual testing particularly) resources. Existing regression testing methods, such as test case selection, test case prioritization and test suite reduction [5] attempt to make regression testing more cost-effective. In this paper we focus on the (selection and) prioritization method(s).

To prioritize the best test cases, one has to set specific characteristics to create an ordering on the test cases. The most direct characteristics usually are code or requirements change coverage. However, in industry other criteria are used as well. In the work reported in [6], the authors proposed some criteria to prioritize potentially relevant test cases. In that work, real test data and criteria was used to illustrate the feasibility of that idea in an industrial context². The proposed criteria were:

1. Number of executions of a test case: how much executions a test was done in previous cycles. The criterion is given by $crit_1 = executionsTC^{-1}$;
2. Test case failure status: This is the failure rate of the total number of executions of a particular test case.

¹Motorola Mobility is our industrial partner

²The work [6] was result of a previous (in the 2004-2008 period) research partnership between Centro de Informática and Motorola Mobility.

This criterion ($crit_2 = failuresTC/executionsTC$) values tests that were most effective in the past;

3. Number of unique failures found: A same test case may have failed countless times and revealing the same defect repeatedly. In this case, a single Change Request (CR) is opened and assigned to this test. This criterion ($crit_3 = CRs$) takes into account the defects found per test not counting repetitions
4. Regression level: each test case is classified according to a code called regression level, which can vary from 1 to 5. The last criterion

$$crit_4 = regressionLevelPoints_n$$

takes this into account, where $regressionLevelPoints_n$ are the points previously assigned to a test whose regression level is n .

When using a weighted ranking function, based on assumed relevant test case criteria, the challenge is to find the best criteria and weights to achieve the best ordering.

In [6] it is shown that by using certain weights on these four criteria, one can create a regression testing campaign automatically by taking the top test cases from the new ordering. Unfortunately, the work [6] left to the test architects the task to find the weights. And because these weights are not searched exhaustively, but proposed manually by guess, the ranking is not the best one in general.

Therefore, our main contributions in this paper are.

- Propose two additional test prioritization criteria (introduced in Section 4) to be considered in the ranking function;
- An optimization model to find the best weights for a ranking function;
- An embedding of the proposed optimization model in the Z3 SMT solver [2]³;
- A semi-automatic calibration strategy based on subsets of test architects choices and the optimization model.

³<http://rise4fun.com/Z3/> (Opt library)

This paper is organized as follows. Section 2 presents our proposed Mathematical Optimization Model. In Section 3 we briefly describe the Z3 tool by showing how to embed the model presented in Section 2 in its language. Section 4 describes some experiments we performed in our proposed model using real data from Motorola Mobility. Section 5 considers our conclusions and future work.

2 Mathematical model

In the work reported in [6], the authors propose a ranking function based on four criteria as presented in Section 1. The ranking function of a test is calculated using the weighted average of the standard points.

$$r_{\vec{w}}^{\vec{c}} = \left(\sum_{i=1}^4 crit_i \cdot weightCrit_i \right) / \left(\sum_{j=1}^4 weightCrit_j \right)$$

where $crit_i$ is the normalized scores on the criteria i and an element of the vector \vec{c} . And $weightCrit_j$ is the weight of criteria j and an element of the vector \vec{w} .

In this section we present a generalization of the work reported in [6] using a Mathematical Optimization Model. Without loss of generality, we do not consider the denominator ($\sum_{j=1}^4 weightCrit_j$). This is important in Section 3 because the ranking function then becomes decidable and solvable by Z3.

Suppose that we have K criteria. Thus a $L \times K$ matrix (the criteria matrix) named $C_{L \times K}$ is a collection of normalized numbers corresponding to each criterion (L corresponds to the amount of records of some entity. In our case, we have L test cases).

As for each criterion we have also an associated weight then we have a K vector of weights. This vector \vec{w} can be set manually, as in [6], or semi-automatically as presented here. Thus, for us, this vector ($\vec{w}_K = (w_1, w_2, \dots, w_K)$) belongs to our set of unknowns.

The ranking function $\vec{r}_{\vec{w}}^{\vec{c}} = (r_1, \dots, r_K)$ we use in this paper (a K vector r_1, \dots, r_K) is given by.

$$\vec{r}_{\vec{w}}^{\vec{c}} = C_{L \times K} \times \vec{w}_K$$

In [6], the authors assumed that certain weights were more important than others by adjusting the weights manually. In our proposal we intend to find such weights by solving a mathematical problem. To do so we use some test case selections from our test architect experts as input to define an objective function.

To show how we transform this problem into an optimization problem, suppose that *Suite* is the set of all test cases and *Chosen* is the set of test cases that a test architect has selected ($Chosen \subseteq Suite$). Our ranking function is used in such a way that if a test case tc_i belongs to *Chosen*, then its ranking function value r_i must be greater than all others r_j ($i \neq j$), corresponding to not chosen test cases.

As the selected test cases (*Chosen*) cannot be explained using all the criteria we consider, we employ an optimization model that tries to use the majority of the elements of

Chosen to satisfy the maximum number of criteria (A criterion is representative whether its weight is greater than zero). Thus we try to maximize the set of test cases selected by the test architects considering the set *Chosen* as our best reference. Suppose that $(\dot{t}c_1, \dots, \dot{t}c_k)$, where $tc_i \in Chosen$ ($1 \leq i \leq k$), is a boolean vector corresponding to the test cases chosen by a test architect. These boolean elements are interpreted such that if $\dot{t}c_i$ is true then tc_i is *elect*ed to belong to the final selection. Otherwise, it is discarded. Hence, our optimization problem can be stated as.

$$\begin{aligned} &\mathbf{Max} \quad (\dot{t}c_1, \dots, \dot{t}c_k), \mathbf{Subject\ to} \quad r_i > r_n, \text{ if } \dot{t}c_i \\ &\mathbf{where} \quad tc_n \in (Suite \setminus Chosen) \cup \{tc_r \mid 1 \leq r \leq k \wedge \neg \dot{t}c_r\} \end{aligned}$$

In the above optimization problem characterization, the maximization of $(\dot{t}c_1, \dots, \dot{t}c_k)$ means to get a fully k -vector of truth values whenever possible, but accepting a partial vector as well. The restrictive part states that if a chosen test case tc_i is elected by this model, then its ranking (r_i) has to be greater than all other rankings of the not elected ($\{tc_r \mid 1 \leq r \leq k \wedge \neg \dot{t}c_r\}$) as well as not chosen ($Suite \setminus Chosen$) test cases.

It is worth noting that, as we will see in Section 4, extreme cases can happen. If none or a single test case can be elected by the above model, then the test architect's choices cannot be safely described by the criteria used in this paper.

Another interesting point is that if a weight is unnecessary, our optimization model simply sets its value to zero.

3 The Z3 SMT Solver

Z3 is an SMT solver from Microsoft Research [2]. It is targeted at solving problems that arise in software verification and software analysis. Consequently, it integrates support for a variety of theories.

Recall from Section 2 that we have weights. In our Z3 encoding, weights are described as constants (unknowns to be instantiated by Z3). For instance, our four criteria become.

```
(declare-const w1 Int) ...
(declare-const w4 Int)
```

The ranking function, corresponding to each element r_i of Section 2, can be stated in Z3 very easily as follows.

```
(define-fun f ((c1 Real) (c2 Real)
              (c3 Real) (c4 Real)) Real
  (+ (* w1 c1) (* w2 c2)
     (* w3 c3) (* w4 c4) )
```

The previous function is used to compute each ranking r_i , where $c1, \dots, c4$ correspond to the data from our industrial partner as will be clarified as in what follows.

Test cases are described in Z3 via two declarations. First we name each test case by declaring a named constant. For example, a test case 1 becomes the following constant declaration (its corresponding ranking value r_1).

```
(declare-const r1 Real)
```

As each test case is associated with four instances of the chosen selection criteria, we state an assertion for each of them as is illustrated in what follows for the test case 1.

```
(assert (= r1 (f 0.06 0.0 0.02 0.5)))
```

As Z3 does not have a min/max operator over sets, we capture this constraint by an indirect way of counting how many chosen test cases can be elected. First we create a $\{0, 1\}$ counter (from the boolean vector but this allows us to sum counters) for each chosen test case. For example, suppose test case 17 was chosen.

```
(declare-const F1TC17 Int)
(assert (or (= F1TC17 0) (= F1TC17 1)))
```

As test case 17 can be elected or not, we use a Z3 conditional construct (*ite*) as follows. In this Z3 encoding, the part starting with *and* occurs if *F1TC17* equals to 1. Otherwise, there is no constraint over test case 17 (*true*).

```
(assert (ite (= F1TC17 1)
            (and (> TC17 TC2) ...) true))
```

Finally we maximize the elected test cases (As the counters have at most the value 1, our full boolean vector of Section 2 means this sum is the maximum possible one).

```
(maximize (+ ... F1TC17 ...))
```

4 Experiments on Automatic Prioritization

During the development of this work, we identified two additional criteria of interest.

5. Creation date: From discussions with the test architect, we noted that newer tests can find a greater number of failures during testing. So, $crit_5 = creationTC^{-1}$ where *creationTC* is the number of days from the date of test case creation to the present time.
6. Date of the last execution: Like the previous criterion, the date of the last execution also increases the scores. Thus, $crit_6 = LastExecutionTC^{-1}$ where *LastExecutionTC* is the number of days from the date of the last execution of the test case to the present time.

In the following experiments we used a Core i5 1,7 GHz MacBook Air with 8 GB RAM, running El Captain 10.11.3 OS. Two test architects helped us to perform the experiments on three test suites creation. The first two experiments (see Tables 1 and 2) involving 80 test cases and the last one (see Table 3) with 427 test cases. In these tables, the test cases in bold face are common between the two test architects choices. Each table has four columns:

- **Chosen**: Number of test cases chosen manually;

Chosen	Elected	Criteria	Time	Match
12	1, 19	1, 2, 5, 6	4m 21.5s	16.7%
25	None	1, 2, 3, 5, 6	11.8s	0%

Table 1. 1st experiment (80 test cases)

Chosen	Elected	Criteria	Time	Match
20	1, 2, 3, 65	1, 2, 5, 6	1m 31.1s	20%
22	1, 2, 3, 65	1, 2, 5, 6	2m 22.8s	18.2%

Table 2. 2nd experiment (80 test cases)

- **Elected**: Test cases chosen by our optimization model;
- **Criteria**: Criteria found by the optimization model;
- **Time**: This is the effort needed to solve the Z3 model;
- **Match**: Elected divided by chosen test cases.

In Table 1 the intersection of **Chosen** test cases between the architects is of 27.6% (They both chose 29 test cases in total where 8 have been chosen by both: $8/29 = 27.6\%$). This low intersection in general occurs because the selection is manual and there is a short time to perform this (Just from their opinion, we get the rate as reading and analyzing 400 test cases to elect 100 test cases in 1 hour. This means electing a test case in about 2.25 seconds). This can be very error-prone. And this was captured by Z3 where one experiment elected 2 test cases and the other none at all.

Table 2 exhibited a better outcome between architects. Common choices were around 45% of all test cases involved. Four test cases were elected by Z3 and they were the same for both architects choices.

In Table 3 we have the most expensive experiment for using Z3. A total of 427 test cases with 128 selected test cases for one test architect and 195 for the other (And an intersection of 54 choices or 20%). In view of its size, the columns **Chosen** and **Elected** were only filled with the total amount of test cases.

The best percentage match obtained by [6] was 7.27% using four criteria. We get a 20% with six criteria. From [6], even with the low percentage match, the ranked tests reveals similar bug reports when compared to manual selections. Thus, this seems to indicate that: (i) the ranking is indeed relevant; (ii) further criteria can increase the percentage match; and (iii) Z3 can outperform human weight search (20% versus 7.27%). Besides, we cannot fully match architects choices because their selection is not systematic and use information beyond the criteria used here.

Chosen	Elected	Criteria	Time	Match
195	12	1, 2, 5, 6	2322m 4s	6.1%
128	1	1, 2, 5, 6	1631m 4s	0.8%

Table 3. 3rd experiment (427 test cases)

Threats to validity Our technique depends on three main resources (test selection criteria, test criteria data and test architect choices). The most sensible input is the test architect choices. We saw in the previous section that when there is a representative common choice between test architects, the criteria is found more easily. However, as reported in Table 1, for diverging choices, the six criteria used in this paper sometimes are not enough to describe them.

Internal threats can be test selection criteria and test criteria data. As test criteria data change over time, it is possible that test selection criteria cannot be always the same. But as our Mathematical model is general enough we can simply propose new test selection criteria that the Z3 solver reports us with the right criteria that describe the data.

5 Conclusion

This work has shown that it is feasible, although costly, to automatically balance a ranking function by formulating the work reported in [6] as an optimization problem. As such an automatic balancing is supposed to be used in a well-spaced period of time, the effort taking by Z3 can be affordable. Besides, after knowing the more adequate criteria, their use in practice is almost inexpensive because it is just an arithmetic calculation.

In our experiments, we have used the criteria proposed in the work reported in [6] and added two other criteria. This revealed an increase in the amount of test cases the ranking function can match with the test cases chosen by the test architects. We can also observe that even with the additional two criteria we considered here, the ranking does not fit the human selection with a high percentage match. This suggests investigating additional criteria that can be used in practice. For instance, information retrieval [7] also uses ranking functions based on keyword indexing and frequency. We intend to consider this new criterion as one of our future work.

The work reported in [1] addresses the problem of determining the next set of releases in the course of software evolution in an industrial setting using an optimization problem formulation. It employs simulated annealing and a greedy algorithm to solve the optimization problem and compare both approaches. In [8], the authors use a similar approach but solving the optimization problem using binary constrained particle swarm optimization (BC-PSO). The work reported in [3] follows a non-orthodox approach to test suite reduction. That work proposes FLOWER, which leverages the Ford-Fulkerson method to compute maximum flows and Constraint Programming techniques to search among optimal flows. In our work we also have an optimization problem formulation, but we employ an SMT solver instead of search-based specific algorithms and our goal is to find the weights and not the ranking per se. Maybe the work that is closest to ours in the sense of using off-the-shelf solvers

is [4]. But this work focuses on test suite minimization, removing redundant test cases based on fault coverage, which is a bit different of our goal. As in general all works implement their own search algorithms whereas we have reused an off-the-shelf SMT solver, we intend to investigate the advantages/disadvantages of doing this in future work.

As our experiments exhibited a considerable variation in execution time from one test architect choice to the other, we intend to investigate this as future work.

Another future work we intend to pursue based on the feedback of our test architect is to automatically identify when a test case is a prefix of another test case. One hidden criteria used in practice is to employ just the greatest test case. This task needs a natural language processing as well as a notion of phrase dependency.

Acknowledgments We thank Alice Arashiro, Viviana Toledo, and Lucas Heredia from Motorola Mobility, and Juliano Iyoda for suggestions on the paper. This research is supported by Motorola Mobility.

References

- [1] P. Baker, M. Harman, K. Steinhofel, and A. Skaliotis. Search based approaches to component selection and prioritization for the next release problem. In *ICSM*, pages 176–185, Sept 2006.
- [2] L. De Moura and N. Bjørner. *Z3: An efficient smt solver*. In *14th TACAS*, pages 337–340, Berlin, Heidelberg, 2008. Springer-Verlag.
- [3] A. Gotlieb and D. Marijan. Flower: Optimal test suite reduction as a network maximum flow. In *ISSTA*, pages 171–180. ACM, 2014.
- [4] H.-Y. Hsu and A. Orso. Mints: A general framework and tool for supporting test-suite minimization. In *Software Engineering, 2009. ICSE 2009. IEEE 31st International Conference on*, pages 419–429. IEEE, 2009.
- [5] J.-H. Kwon, I.-Y. Ko, G. Rothermel, and M. Staats. Test case prioritization based on information retrieval concepts. 1:19–26, Dec 2014.
- [6] J. Mafra, B. Miranda, J. Iyoda, and A. Sampaio. Test case selector: Uma ferramenta para seleção de testes (in portuguese). In *SAST*, pages 1–10, 2009.
- [7] R. K. Saha, L. Zhang, S. Khurshid, and D. E. Perry. An information retrieval approach for regression test prioritization based on program changes. In *37th ICSE (vol. 1)*, pages 268–279. IEEE Press, 2015.
- [8] L. Souza, R. Prudêncio, F. Barros, and E. Aranha. Search based constrained test case selection using execution effort. *Expert Systems with Applications*, 40(12):4887 – 4896, 2013.

Practical Combinatorial Testing Approaches: A Case Study of a University Portal Application

Mary Frances Moore and Sergiy Vilkomir

Department of Computer Science
East Carolina University
Greenville, NC 27858 USA
{moorema, vilkomirs}@ecu.edu

Abstract—Time and quality are important factors when determining the proper approach for software testing. A software program can often be used in various environments (different platforms, operating systems, browsers, networks, etc.) and require thorough testing to provide high quality and reliability in different configurations. Combinatorial testing is an effective approach to testing hardware and software configurations. However, testing resources are often restricted in real practice. Because business goals require different testing methods, there is no best one-size-fits-all testing approach. For this reason, we experimentally investigated and analyzed several combinatorial approaches based on Each Choice and pairwise methods (with and without the consideration of operational profiles) through the testing of an Adviser Scheduling application located in a university web portal. Test sets with various configurations were generated according to six different combinatorial strategies. The Advanced Combinatorial Testing System (ACTS) tool, which was provided by the National Institute of Standards and Technology (NIST), was used to generate pairwise test sets automatically. The case study software application was retested for each of the proposed testing approaches, and the results were compared after taking into account the number of test cases and the corresponding detected faults. Based on this analysis, we provide recommendations for the selection of testing approaches to align with different business goals. The recommendation chosen for the university web portal application allowed for improved quality and reduced time for software testing.

Keywords—Combinatorial testing; pairwise; Each Choice; Operational Profile

I. INTRODUCTION

A software program can often be used in various environments, such as different platforms, operating systems, browsers, networks, etc., and it requires comprehensive testing for many configurations to provide high quality and reliability. One of the best approaches in this situation is combinatorial testing [1, 2], which has been confirmed to be practical and effective [3-6].

Combinatorial t -way testing requires that any combination of values of any t testing parameters or configuration items should be included in some test case. This type of testing is often used for $t=1$ (Each Choice testing) or $t=2$ (pairwise testing) [7, 8]. Each Choice covers all values of all parameters, but it does not consider combinations of values. Pairwise testing covers all values *and* combinations of each value with

all others, i.e., it covers all pairs of values. A larger value of t increases the effectiveness of t -way testing, but this requires more test cases, obliging testers to compromise between desirable effectiveness and available testing resources.

Testing resources (time, money, and human resources) are often restricted in real practice. A company's business goal might include improving the quality or effectiveness of testing while keeping the same degree of testing or even reducing the number of test cases, while still maintaining the appropriate level of testing quality. Because different business goals require different testing approaches, there is no "best" testing approach. Sets of different approaches should be considered to select one suitable for the current situation and the specific business goals.

In real practice, some configurations are more common than others. For example, when a particular software application is accessed by many users, Internet Explorer may be used more often than Chrome, Windows 8 more often than Windows Vista, etc. This can be described using an operational profile, which is a quantitative characterization of how a system or software will be used [9]. To achieve trustworthy test results, software testing should be performed according to the operational profile, namely, the proportion of tests for different configurations should approximately reflect the occurrences of these configurations in the software's real usage [10, 11]. However, combinatorial approaches treat all testing configurations equally. In order to reflect the operational profile, these approaches should be modified and the number of configurations extended.

In this paper, we analyzed several combinatorial testing approaches based on Each Choice and pairwise methods, with and without consideration of an operational profile. The paper is organized as follows: Section II explains the organization of our investigation and the proposed testing approaches. The Advisor Scheduling application, used as a case study for applying combinatorial testing, is described in Section III. Section IV contains detailed information on test configurations that were generated according to different combinatorial testing methods. Section V provides experimental testing results of these configurations and analyzes the effectiveness of the proposed approaches. The conclusions are presented in Section VI.

II. TESTING APPROACHES AND ORGANIZATION OF THE INVESTIGATION

Our experimental investigation included retesting the Advisor Scheduling application using six combinatorial approaches based on Each Choice and pairwise testing. This application, used as a case study and described in the next section, has already been deployed and tested in depth but without using combinatorial approaches. The first version of this application, which contained all the bugs found and removed in the original testing phase, was retested. Our results were compared to the original test results, taking into account the number of test cases and the corresponding bugs found for each combinatorial approach. Our goal was to find an approach comparable in size to the number of original test cases while providing greater effectiveness in fault detection.

We investigated the following six approaches:

- Each Choice
- Each Choice with consideration of the operational profile
- Each Choice with additional parameters
- Pairwise
- Pairwise with consideration of the operational profile
- Pairwise with additional parameters

The “pure” Each Choice and pairwise approaches used the same configuration parameters and their values as the initial testing. However, Each Choice required significantly fewer test cases. Pairwise required approximately the same number of test cases as the initial testing but provided much better coverage of parameter value combinations. Taking into consideration the operational profile, we suggested using the same method for Each Choice and pairwise. Usually, the precise reflection of operational profiles requires using probabilistic models and a large number of test cases [12], but this was not possible for practical testing in our case study. An approximate approach to consider operational profiles has been suggested by Kuhn et al. [1, pp. 61-64] by adding additional values of parameters. We suggested another simple approximate method of adding one or two additional test cases into the test sets with the most frequently used parameter values. This made the distribution of values in test sets closer to the real operational profile.

Because combinatorial approaches require a small number of tests, it was possible to add additional configuration parameters and still have a comparable number of test cases compared to those in the original testing. In turn, new parameters allowed testing new configurations that were not tested originally. Detailed analysis of the generated test cases (Section IV) and the results of testing for different approaches (Section V) allowed for the provision of practical recommendations for the selection of testing approaches to align with different business goals.

III. A CASE STUDY: ADVISOR SCHEDULING APPLICATION

A. Description

Many universities use a web portal to provide students, faculty, and staff with access to valuable data and applications

within one location. Access to a portal is usually granted with university credentials. These credentials validate the access the user should have. For instance, when an employee logs into a portal, he or she may have access to applications such as Pay History and Workplace Training. However, when a student logs in, he or she would not get access to those applications. Instead, other applications would be available such as GPA Calculator, Grade History, and Course Schedule. Frequently, applications are available to multiple user types, but each user is granted a different level of access based on need. For example, an employee user type could view announcements in an application, but an administrator could add, edit, and delete the announcements. Each application within a portal is customized to fit the needs of all users having access to the application.

Before each application within a portal is released to its users, it must be tested thoroughly to ensure that it works for all user types. Testing configurations should be created to include user access testing, along with other parameters such as browser, operating system, etc. Testing multiple configurations can ensure that the application is ready for all users who will access the application in different ways. This case study reviews the original testing process for an Advisor Scheduling application in a university portal and proposes and implements practical combinatorial testing approaches.

The Advisor Scheduling application is used for advising processes specified by an Academic Advising Collaborative. The application has four main functions: scheduling, accepting appointments, viewing academic information, and processing appointment outcomes. The scheduling feature and processing appointment outcomes feature is accessible by all users of the application. The accepting appointments and viewing academic information features are available only to advisors.

There are three different user permission groups in the Advisor Scheduling application: administrators, advisors, and delegates. Each group is given limited access to the application based on the permission group they have been assigned. For users to access the Advisor Scheduling application, they must first log into the university portal. Once they have done so, using their university credentials, they can find the Advisor Scheduling application in the portal’s Tools section.

The application’s interface allows advisors to manage and perform routine tasks with ease. After opening the application, the advisor will see their calendar, the advisee roster, and outstanding actions. The outstanding actions list contains appointment requests created either by delegates within the Advisor Scheduling application or by students using the student scheduler application (which is a separate entity from the Advisor Scheduling application). When an advisor accepts or rejects an appointment displayed in their Outstanding Actions list, the calendar and advisee roster is updated accordingly. After accepting an appointment, the calendar immediately displays the appointment in its allotted time slot. At the same time, the advisee roster adds the student for the newly created appointment to the top of the roster. If rejected, the appointment is removed from the Outstanding Actions list and not added to the calendar. Fig. 1 displays a portion of an adviser’s calendar with student appointments.



Figure 1. Adviser Scheduling Application – Redacted for Confidentiality

B. The Original Testing Phase

When originally testing the Adviser Scheduling application, one test set consisting of 11 configurations of parameters was used, with a total of 121 tests completed for the application’s 21 test cases. The configuration parameters and their values included:

- Browser: IE, Chrome
- User type: Faculty, Administrator, Delegate
- Operating System: Windows 7, OS X
- Network: On Campus–Secured, On Campus–Unsecured, Off Campus

These configurations were determined by the testing time allotted as the project’s deadline approached, along with tester knowledge of the portal and experience of the types of bugs normally found. The original testing procedure was found to be adequate and thorough; however, there were not enough resources to provide mobile testing. Table 1 shows the configurations of the parameters listed above that were used in testing and the number of tests completed for each configuration.

TABLE I. ORIGINAL TESTING CONFIGURATIONS AND NUMBER OF TESTS COMPLETED

Config	Browser	User Type	Operating System	Network	Tests
1	IE	Faculty	Win 7	On Campus–Secured	21
2	IE	Admin	Win 7	On Campus–Secured	10
3	IE	Delegate	Win 7	On Campus–Secured	12
4	IE	Faculty	Win 7	On Campus–Unsecured	7
5	IE	Faculty	Win 7	Off Campus	7
6	Chrome	Faculty	Win 7	On Campus–Secured	21
7	Chrome	Admin	Win 7	On Campus–Secured	10
8	Chrome	Delegate	Win 7	On Campus–Secured	12
9	Chrome	Faculty	OS X	On Campus–Secured	7
10	Chrome	Faculty	OS X	On Campus–Unsecured	7
11	Chrome	Faculty	OS X	Off Campus	7

C. The Operational Profile

According to the data collected by the Academic Technologies team, 42 percent of users access the Adviser Scheduling application with Internet Explorer. Faculty are the only user type to have access to all 21 of the application’s test cases, while the other user types perform auxiliary functions, making the faculty user type the most common. Due to the fact that the standard operating system provided to university faculty is Windows 7 connected to the on-campus secured network, they are most commonly used to access the application. In addition, the data collected by the Academic Technologies team suggested that 16 percent of user traffic comes from mobile devices; however, that percentage has risen and will continue to rise due to mobile device and tablet popularity and convenience. When testing with the Each Choice and pairwise testing methods, the operational profile data can be considered while determining the test sets as a way of tailoring the test sets to the user majority.

IV. APPLICATION OF THE TESTING APPROACHES

The two methods being explored through re-testing of the Adviser Scheduling application are Each Choice testing and pairwise testing, both with and without consideration of the application’s operational profile. The Each Choice testing method provides test sets that allow for a significant reduction in the amount of testing while retaining or improving the testing quality. The pairwise testing method provides test set sizes similar to or slightly larger than the original testing approach.

With both the Each Choice and pairwise testing methods, variations can be made to the test sets to tailor the methods to testing needs. Three sets of testing approaches were used for both methods to test the Adviser Scheduling application. For each testing method, at least one approach aims to reduce the number of required test configurations, one takes the application’s operational profile into consideration, and one aims to increase the test coverage and number of detected bugs.

A. Each Choice Testing Approaches

The three test sets (Tables II, III, and IV) were created using the Each Choice testing method. The goal of the results from these tables is to maintain testing quality while significantly reducing the number of required tests.

Each Choice Test Set 1: This Each Choice testing table includes the application’s original testing parameters shown in Table II. The table covers every parameter tested originally at least once using three configurations.

TABLE II. EACH CHOICE TEST SET 1

Config	Browser	User Type	Operating System	Network	Tests
1	IE	Faculty	Win 7	On Campus–Secured	21
2	Chrome	Admin	OS X	On Campus–Unsecured	10
3	IE	Delegate	Win 7	Off Campus	12

Each Choice Test Set 2: With the resources provided by the Academic Technologies team, the operational profile of the Adviser Scheduling application was determined and considered when testing with the Each Choice and pairwise testing methods. The data in Table III include the most common parameter values based on the operational profile. This extra configuration was added to Test Set 1 to consider the application’s operational profile, making Test Set 2 consist of four configurations.

TABLE III. EACH CHOICE TEST SET 2

Config	Browser	User Type	Operating System	Network	Tests
4	IE	Faculty	Win 7	On Campus–Unsecured	21

Each Choice Test Set 3: This Each Choice testing table introduces a new parameter and values and consists of five configurations (see Table IV). The aim of this test set is to reduce the number of required tests while increasing the number of bugs found by testing mobile devices, which were not a part of the application’s original test range.

TABLE IV. EACH CHOICE TEST SET 3

Co nf	Brow- ser	User Type	Oper. System	Network	Mo- bile	Tests
1	IE	Faculty	Win 7	On Campus–Secured	No	21
2	Chrome	Admin	Win 10	On Campus–Unsecured	No	10
3	Firefox	Delegate	OS X	Off Campus	No	12
4	Safari	Faculty	iOS	On Campus–Secured	Yes	21
5	Stock	Admin	Android	On Campus–Unsecured	Yes	21

B. Pairwise Testing Approaches

To generate the test sets used for pairwise testing in this case study, the Advanced Combinatorial Testing System (ACTS) tool was used. Provided by the National Institute of Standards and Technology, ACTS is a free tool that assists users in generating *t*-way combinatorial test sets. Prior to using the ACTS tool, the user must identify the parameters that will be used in testing, along with their values and associated constraints. The tool eliminates any combinations that violate constraints between parameters as they were configured while creating the system, and then it allows the user to build and view the test set. For the Adviser Scheduling application case study, the ACTS tool was used to create pairwise (2-way) test sets for the application’s 21 test cases. Three test sets were created using the ACTS tool.

Pairwise Test Set 1: This test set includes the parameters used in the original testing phase to show that less testing could be performed with a much higher level of coverage. This test set includes nine configurations of the original testing

parameters, shown in Table V, which is two configurations fewer than used originally.

TABLE V. PAIRWISE TEST SET 1

Config	Browser	User Type	Operating System	Network	Test
1	Chrome	Faculty	OS X	Off Campus	21
2	IE	Faculty	Win 7	On Campus–Secured	21
3	Chrome	Faculty	Win 7	On Campus–Unsecured	21
4	IE	Admin	Win 7	Off Campus	10
5	Chrome	Admin	OS X	On Campus–Secured	10
6	IE	Admin	Win 7	On Campus–Unsecured	10
7	Chrome	Delegate	OS X	Off Campus	12
8	IE	Delegate	Win 7	On Campus–Secured	12
9	Chrome	Delegate	OS X	On Campus–Unsecured	12

Pairwise Test Set 2: The second test set includes the parameters used in the original testing phase, the same nine configurations as in Test Set 1, and an additional two configurations that are derived from the operational profile. These additional two configurations are shown in Table VI.

TABLE VI. PAIRWISE TEST SET 2

Config	Browser	User Type	Operating System	Network	Tests
10	IE	Faculty	Win 7	On Campus–Unsecured	21
11	IE	Faculty	Win 7	Off Campus	21

Pairwise Test Set 3: The third test set introduces more parameters and parameter values to broaden the testing scope with the intent to find new bugs. This test set includes 19 configurations, listed in Table VII, based on five parameters. The new parameter introduced in this set is “Mobile” with values “Yes” and “No.” Several new parameter values were added for parameters “Browser” and “Operating System.”

V. RESULTS OF THE INVESTIGATION

Fig. 2 and Table VIII display the results of the investigation, comparing the number of tests completed for each test set with the number of bugs found. In addition, Table VIII compares the number of parameters and their values and the number of configurations tested per test set. A total of 43 tests were performed while testing Each Choice Test Set 1. This test set provided the same quality of testing as the original procedure, while reducing the number of tests from 121 to 43, essentially cutting the number of tests by 65%. While testing Each Choice Test Set 2, 64 tests were performed that detected the same 53

bugs found during the original testing. Each Choice Test Set 2 consisted of 21 more test cases than Each Choice Test Set 1 to include consideration of the operational profile. Each Choice Test Set 3 consisted of 85 tests. This test set introduced testing coverage for mobile devices. A total of 75 bugs were found during testing. These bugs consisted of the same 53 bugs detected during the original testing of the Advisor Scheduling application, in addition to 22 new bugs that were specific to mobile devices and mobile browsers.

TABLE VII. PAIRWISE TEST SET 3

Co nf	Brow- ser	User Type	Oper. System	Network	Mo- bile	Tests
1	IE	Admin	Win 7	On Campus- Unsecured	No	10
2	IE	Delegate	Win 10	Off Campus	No	12
3	Chrome	Faculty	Win 7	On Campus- Secured	No	21
4	Chrome	Admin	Win 10	On Campus- Unsecured	No	10
5	Chrome	Delegate	OS X	Off Campus	No	12
6	Chrome	Faculty	Android	On Campus- Unsecured	Yes	21
7	Chrome	Admin	iOS	On Campus- Secured	Yes	10
8	Firefox	Delegate	Win 7	Off Campus	No	12
9	Firefox	Faculty	Win 10	On Campus- Secured	No	21
10	Firefox	Admin	OS X	On Campus- Unsecured	No	10
11	Firefox	Delegate	Android	On Campus- Secured	Yes	12
12	Safari	Faculty	OS X	Off Campus	No	21
13	Safari	Delegate	iOS	On Campus- Unsecured	Yes	10
14	Stock	Admin	Android	Off Campus	Yes	12
15	Safari	Faculty	iOS	Off Campus	Yes	21
16	IE	Faculty	Win 7	On Campus- Secured	No	21
17	Stock	Faculty	Android	On Campus- Unsecured	Yes	21
18	Safari	Admin	OS X	On Campus- Secured	No	10
19	Stock	Delegate	Android	On Campus- Secured	Yes	12

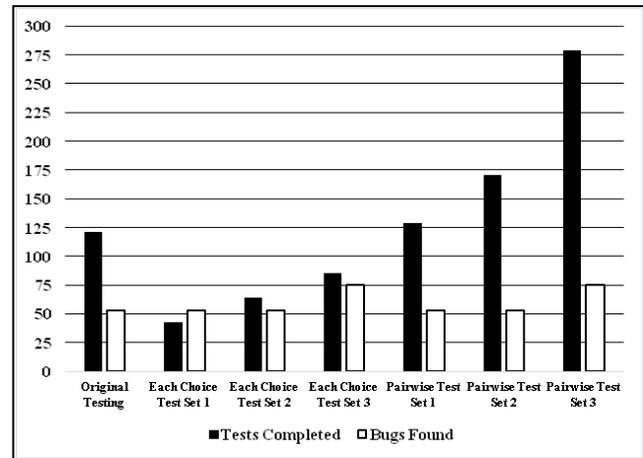


Figure 2. Comparison of the Number of Tests Completed to Number of Bugs Found

TABLE VIII. NUMBER OF PARAMETERS AND CONFIGURATIONS FOR EACH TESTING METHOD

	Test Sets	Para- meters	Para- meter Values	Config. per Test Set	Tests	Bugs
1	Original test	4	10	11	121	53
2	Each Choice Test Set 1	4	10	3	43	53
3	Each Choice Test Set 2	4	10	4	64	53
4	Each Choice Test Set 3	5	16	5	85	75
5	Pairwise Test Set 1	4	10	9	129	53
6	Pairwise Test Set 2	4	10	11	171	53
7	Pairwise Test Set 3	5	16	19	279	75

Pairwise Test Set 1 consisted of 129 tests. This test set provided the same quality of testing as the original procedure. However, it required slightly more test cases but fewer test configurations. The amount of testing was approximately the same as it was for original testing, but the provided level of coverage was much better. While testing Pairwise Test Set 2, 171 tests were performed that detected the same 53 bugs as found during the application’s original testing. Pairwise Test Set 2 consisted of 42 more test cases than Pairwise Test Set 1 to include consideration of the operational profile. Pairwise Test Set 3 consisted of 279 tests to include testing coverage for mobile devices. A total of 75 bugs were found during testing, as with Each Choice Test Set 3. Each Choice Test Set 3 provided the same quality of testing as Pairwise Test Set 3, while reducing the number of tests from 279 to 85. While in our case study pairwise testing did not demonstrate additional benefits when compared with Each Choice, in other situations results could be different.

VI. CONCLUSIONS

In conclusion, we found that the original testing of the application was thorough, but while good enough to detect the same bugs as the pairwise and Each Choice approaches, it was not systematic and did not allow time for mobile testing. The purely pairwise method and Each Choice method did not reveal any new bugs but reduced the number of configurations. Reducing these also reduces the effort required in testing. Although pairwise testing requires completing more tests than Each Choice, the latter cannot produce better results than pairwise testing, although it can reduce the testing time. The Each Choice testing method allowed for testing an extra parameter (mobile) without exceeding the time and resource restriction as originally faced in testing. While no new bugs were found when considering the operational profile, there is greater confidence in the test results that the application is ready for use by its main audience.

In this case study, pairwise was not the most beneficial testing method, but it is usually considered to be the better testing alternative (compared to Each Choice) when sufficient resources are available. Our case study is limited to one application and does not intend to compare these approaches in detail. However, even this one example demonstrates how practical combinatorial testing approaches can minimize the number of test cases and/or maximize the number of detected faults.

With the results of fewer tests, new configurations, and better detection rates, this study shows that the Each Choice testing method with the inclusion of the mobile device parameter can be implemented as the preferred testing approach for applications within the university's web portal.

ACKNOWLEDGMENT

This work was performed under the following financial assistance award 70NANB15H217 from the U.S. Department of Commerce, National Institute of Standards and Technology.

REFERENCES

- [1] D. R. Kuhn, R. Kacker, and Y. Lei, *Introduction to Combinatorial Testing*, Chapman and Hall/CRC, 2013, 341 pages.
- [2] D. R. Kuhn, R. Kacker, Y. Lei, and J. Hunter, "Combinatorial software testing," *IEEE Computer*, vol. 42, no. 8, August 2009, pp. 94–96.
- [3] P. J. Schroeder, P. Bolaki, and V. Gopu, "Comparing the fault detection effectiveness of n-way and random test suites," *Proceedings of the 2004 IEEE International Symposium on Empirical Software Engineering (ISESE'04)*, 19–20 August 2004, Redondo Beach, CA, USA, pp. 49–59.
- [4] D. R. Kuhn, Y. Lei, and R. Kacker, "Practical combinatorial testing: Beyond pairwise," *IT Professional*, 10.3, 2008, pp. 19–23.
- [5] S. Vilkomir, K. Marszalkowski, C. Perry, and S. Mahendrakar, "Effectiveness of Multi-Device Testing Mobile Applications," *Proceedings of the 2nd ACM International Conference on Mobile Software Engineering and Systems (MobileSoft 2015)*, May 16–17, 2015, Florence, Italy, pp. 44–47, in conjunction with the 37th International Conference on Software Engineering (ICSE'15).
- [6] S. Vilkomir, O. Starov, and R. Bhambroo, "Evaluation of t-wise Approach for Testing Logical Expressions in Software," *Proceedings of the IEEE Sixth International Conference on Software Testing, Verification and Validation Workshops (ICSTW 2013)*, 18–20 March 2013, Luxembourg, Luxembourg, pp. 249–256.
- [7] J. Czerwonka, "Pairwise testing in Real World. Practical extensions to test case generators," *Proceedings of 24th Pacific Northwest Software Quality Conference*, Portland, Oregon, October 9–11, 2006, pp. 419–430.
- [8] M. Grindal, J. Offutt, and S. F. Andler, "Combination testing strategies: a survey," *Software Testing, Verification and Reliability*, vol. 15, no. 3, March 2005, pp. 167–199.
- [9] J. Musa, "Operational profiles in software-reliability engineering," *IEEE Software*, 10.2, 1993, pp. 14–32.
- [10] C. Smidts, C. Mutha, M. Rodríguez, and M. J. Gerber, "Software testing with an operational profile: OP definition," *ACM Comput. Surv.* 46, 3, Article 39, February 2014, 39 pages.
- [11] P. A. Brooks and A. M. Memon, "Automated gui testing guided by usage profiles," *Proceedings of the 22 IEEE/ACM international conference on Automated software engineering*, November 5–9, 2007, Atlanta, Georgia, USA, pp. 333–342.
- [12] J. A. Whittaker and M. G. Thomason, "A Markov Chain Model for Statistical Software Testing," *IEEE Trans. Softw. Eng.* 20, 10, October 1994, pp. 812–824.

On Building Test Automation System for Mobile Applications Using GUI Ripping

Chuanqi Tao

Nanjing University of Science and Technology
Nanjing, Jiangsu, China
taochuanqi@njust.edu.cn

Jerry Gao

San Jose State University
San Jose, CA, USA
Taiyuan University of Technology, Taiyuan, China
Corresponding to: jerry.gao@sjsu.edu

Abstract: With the rapid advance of mobile computing technology and wireless networking, there is a significant increase of mobile subscriptions. This brings new business requirements and demands in mobile software testing, and causes new issues and challenges in mobile testing and automation. Current existing mobile application testing tools mostly concentrate on GUI, load and performance testing which seldom consider large-scale concurrent automation, coverage analysis, fault tolerance and usage of well-defined models. This paper introduces an implemented system that provides an automation solution across platforms on diverse devices using GUI ripping test scripting technique. Through incorporating open source technologies such as Appium and Selenium Grid, this paper addresses the scalable test automation control with the capability of fault tolerant. Additionally, maximum test coverage can also be obtained by executing parallel test scripts within the model. Finally, the paper reports case studies to indicate the feasibility and effectiveness of the proposed approach.

Keywords: mobile application testing; test automation; large-scale concurrent testing; GUI ripping

I. Introduction

With the rapid advance of mobile computing technology and wireless networking, there is a significant increase of mobile subscriptions. This brings new business requirements and demands in mobile software testing, and causes new issues and challenges in mobile testing and automation. Mobile Application testing refers to testing activities for native and web applications on mobile devices using well-defined software test methods and tools to ensure quality in functions, behaviors, performance, quality of service and features like mobility, usability, inter-operation ability, connectivity, security and privacy [10][22]. Most of the present research work focuses on providing solutions to the technical problems on mobile app white-box testing methods, GUI-based test technique, and ad-hoc scripting test tools;

GUI ripping has become an effective technology in GUI test automation. Test cases and test execution can both be automated using well-defined GUI ripping models [4][14][19][21]. Current existing mobile app tools mostly concentrate on GUI, load and performance testing which hardly considers large-scale concurrent automation,

coverage analysis, fault tolerance and usage of well-defined models. The existing test automation approaches and tools suffer from several challenges and problems. The first issue is that it is too costly to deal with diversity of mobile test environments on varieties of mobile devices, and the other is the lack of cost-effective method and platforms to support a unified mobile test automation crossing different mobile platforms on diverse devices. Most of existing tools do not analyze the data dependence between GUI components. In summary, there is a lack of mobile test scripting technique to address large-scale concurrent mobile test needs.

This paper intends to develop a system tool that provides a large-scale automation solution using GUI ripping. By incorporating open source technologies such as Appium and Selenium Grid, we plan to address the scalable test automation control. Additionally, maximum test coverage can also be obtained by executing parallel test scripts within the model. Hence, this approach will be useful in numerous mobile app validation and test automation within the industry. The contributions of this paper can be summarized as follows.

- Test automation solution that would provide services for concurrent test automation for multiple mobile devices running on different platforms.
- Dependency analysis report which helps in identifying the dependency existing between the multiple scripts intended to run on different paths.
- Coverage analysis report after the end of each successful test execution.

The paper is structured as follows. The next section presents the background and related work. The system requirement analysis is introduced in Section III. Section IV presents the designed and implemented prototype tool for the proposed approach. Case studies of testing on sample mobile apps are shown in Section V. Conclusions and future work are summarized in Section VI.

II. Background and Related Work

A. Background

The purpose of ripper is to identify maximum number of structural information about the GUI of any android application through the algorithm. Here, the application's windows are opened in depth-first manner. The ripper is capable of extracting all the widgets and its properties from a GUI window. Properties of a window can be size, color, status or window types. GUI windows can be categorized into two types. Model window allows users to fire an event

from within the window like a Save button. Modeless window on the other hand, let users expand a set of commands like Replace command, which doesn't restrict the user's focus to a specific range of events within a window. Thus, ripper abstracts widgets and their properties and generates a GUI tree.

Hierarchical nature of the GUI is represented in a GUI tree. Each node of the GUI tree is called a window or activity [14], [19]. An Event-Flow Graph (EFG) is generated by taking the input as a GUI tree that is the outcome of applying the ripper algorithm on the application. An EFG is a directed graph epitomizing the GUI events. Each node in the EFG is any GUI event. An edge from node *u* to node *v* represents a 'follows' relationship that suggests event *v* can be executed soon after the event *u*. Each event in the EFG will contain the widget ID with which, the matching GUI node can be identified through the GUI tree.

B. Related work

Huang et al. proposed an automated test case generation framework for GUI testing. An accessibility framework known as Microsoft UI Automation (UIA) is used to perform Reverse engineering to generate the event flow graph [1]. Memon et al. discussed GUI test modeling, test generation and replay, and factors for controlling flakiness of generated test cases [2]. The models used for event spacing are state machine and graph models. Extracting, generating and running GUI tests are performed using a tool called GUITAR (GUI Testing frAmewRk). Li et al presented a solution for GUI automation testing for smartphone applications from user behavior models known as AD Automation framework [3]. It supports behavior modeling, automated GUI test case generation, test case simulation and error diagnosis based on UML activity diagrams. Test script library is constructed based on the model and GUI test case generation makes use of these libraries. Also the log analyser performs the error diagnosis after test execution. Amalfitano et al proposed a technique that automates Android mobile applications testing [4]. Amalfitano et al. also presented a tool called AndroidRipper that uses an automated GUI-based technique to test Android apps in a structured manner. It relies on a GUI crawler which crawls through the application GUI to generate test cases. This crawler will then reveal the fault points in the application such as runtime crashes and it can also be used in the regression testing. GUI ripping has become a popular technology in GUI test automation using well-defined GUI ripping models [2, 3]. GUI-based testing has been discussed in numerous papers. For instance, Anand et al. [20] introduced an automated testing approach to validating mobile GUI event sequences for smart-phone apps. Similarly, Amalfitano and his colleagues [21] presented a tool called AndroidRipper that uses an automated GUI-based technique to test Android apps in a structured manner.

Certain research efforts are dedicated to developing tools or frameworks to address limitations in current commercial

tools. For example, JPF-ANDROID is a verification tool that supports white-box testing for mobile apps, and JeBUTi/ME is a tool that offers white-box test coverage analysis based on a conventional CFG-based test model. A few recent research papers focus on GUI-based testing using scripts and GUI event-flow models [20][21].

III. System Requirement Analysis

The main business requirement for the system is test automation which is cost effective and achieves maximum coverage by following the model based testing. We have used open source tools to achieve test automation. GUI Ripper automatically analyses the mobile application by examining the Graphical User Interface (GUI). It uses both standardized exploration strategies such as DFS (Depth First Search) and BFS (Breadth First Search) and random techniques. This results in an abstract GUI model which is also known as GUI tree [19]. An Event Flow Graph can be deduced from this GUI tree graph. Test sequence and test cases are generated based on the Event Flow Graph. This type of model based approach helps to achieve maximum degree of coverage of the application UI.

The description of use case can be illustrated as follows. User accesses the web page and uploads the application which is an *apk* file; A model is generated by the automation server by the GUI ripping process, which the user is able to view in an XML format; The user can also view the dependencies; User gets the tests scripts being generated by the automation server; The test scripts are run by the user with the help of runner component; After the test script execution, user can view the test execution summary; Coverage analysis is provided to the user. Table 1 shows system level functionalities by explaining the basic components available in our system and the corresponding responsibilities. Due to space limitation, other requirements such as non-function, behavior, resource, and etc are not listed here.

Table 1 Several typical function requirement of the system

Components	Responsibilities
Emulator/Mobile Environment Setup	Helps during the mobile/Emulator environment setup at deployment
GUIRipper	Traverses applications' GUI and generate GUI model
EFG	Converts GUI model into event flow graph with XML format
TestGenerator	Identifies the action sequence and generating test scripts based on it.
CoverageAnalyzer	Responsible for test result analysis and coverage analysis.

IV. Design and Implementation for the System

This section presents the detailed software system architecture and explains about the involved components and their relations and connectivity.

System architecture design

Existing mobile testing tools face several challenges and issues. We intend to design and implement a feasible cost

effective solution to model based test automation as depicted in Figure 1. Test automation web application is an interface for users to upload android APK file. It displays model and graph. Users can also view the test execution and test summary using the UI interface through MAC terminal. Test Automation Server takes the uploaded APK file as the input to proceed further with ripping, analyzing dependencies and generating test cases. Test scripts are built based on the grouped scenarios. These test scripts are fed to the test runner module. Test runner passes the test scripts to the selenium grid. With the *appium* nodes running, test scripts are executed on the registered mobile devices and emulators. Model manager stores the path details and generates model in the database and likewise Test Gen and Runner store test scripts and test results respectively.

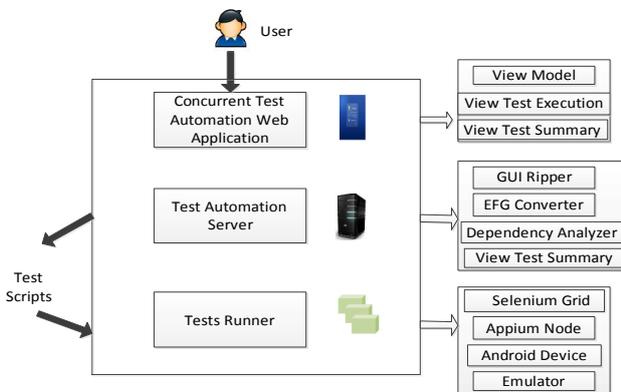


Figure 1 High level infrastructure diagram

Conceptual Framework- To achieve model based test automation, we designed a test automation server. It is a conceptual framework which includes the interaction between all the components of the automation servers. We have included the sub modules of each component. Parser in the diagram is tool which fetches the elements from the under test application. Using this application data in GUI traverse or GUI ripping algorithms we can generate the test model in the model manager component. We also perform Scenario extraction and dependency analysis using this generated model. Dependencies here are among screens, function etc. Based on the analysis of model manager, a Test Gen component generates the action sequence and test scripts. These test scripts are then fed into Runner component where the actual test execution occurs. Finally the Test Analyzer component does the post-test analysis to generate the coverage report and test summary.

Centralized Test Automation Platform- To achieve scalability, we designed a scalable grid node diverse framework. Multiple scripts can run on multiple mobile devices irrespective of the platform and version. Some well-known open source technologies in Selenium grid and Appium are incorporated.

Test Runner Components- As depicted in Figure 2, test runner components constitute of selenium grid, appium nodes, android mobile devices and emulators. Selenium grid

acts as hub and supports large scale distributed testing. It manages multiple nodes, checks for active and inactive grid and updates the status. Multiple appium nodes can be registered to the hub. It supports mobile application automation testing. Multiple devices and emulators can be incorporated for test automation using selenium hub – appium node configurations.

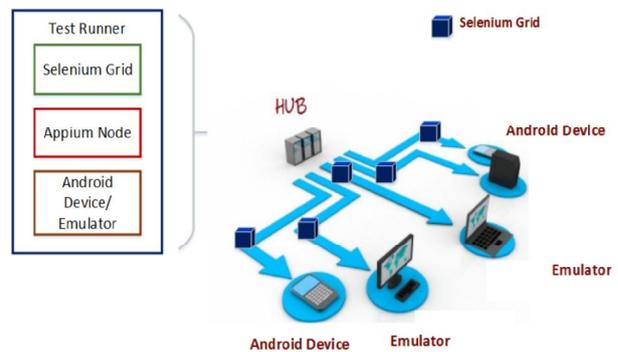


Figure 2 TestRunner component samples

Test Automation Server Components- We have divided our test automation server into four main modules, i.e., GUI Ripper, Dependency Analyzer, Test Gen, Runner to serve various purposes like generating test model, analyzing dependencies, generating testing cases and executing test scripts.

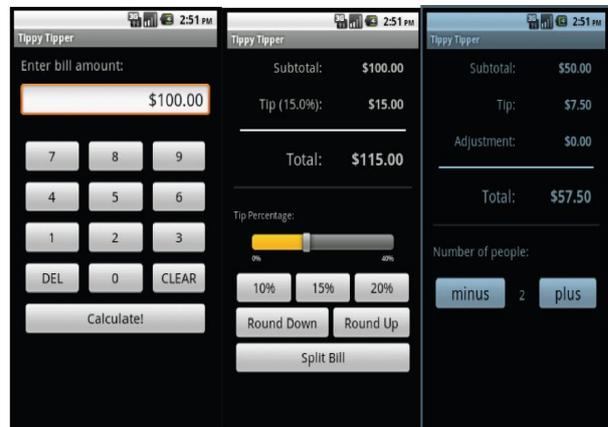


Figure 3 TippyTipper sample android application

V. Case Studies

A. Study object

To apply the developed tool for mobile application GUI testing, we used several realistic mobile applications and systems. Here we choose TippyTipper as the sample android application to illustrate our approach. This is a simple application to calculate the tips for a meal. Figure 3 illustrates the simple functionalities existing in the application. This application has 3 windows. In the first window, it allows the users to enter the bill amount with numeric keypad. The DEL button deletes a rightmost (to the

cursor) numeric number and CLEAR button clears the number entered in the edit text. The CALCULATE button allows the user to move into the second window. Second window displays total bill amount including the calculated tax amount. It also has the progress bar with which you can modify the percentage of tax you want to calculate (0-100), option to roundup and split the bills. On pressing the SPLIT BILLS, the control goes to the third window, which has the option to modify the number of users within the bill has to be shared.

Below subsection shows the detailed approach we followed for automated testing of TippyTipper application.

B. Model generation

We have used the open source GUI ripper algorithm for ripping the GUI. This will result in a file with its GUI components which is basically a GUI tree. This is later fed into the converter, which eventually produces our model as Event Flow Graph.

The purpose of Ripper is to identify maximum number of structural information about the GUI of any android application through the algorithm. Here, the application's windows are opened in depth-first manner. The ripper is capable of extracting all the widgets and its properties from a GUI window. Properties of a window can be size, color, status or window types. GUI windows can be categorized into two types. Model window allows the user to fire an event within the window, like a Save button. Modeless window on the other hand, allows the user to expand a set of commands like Replace command, which doesn't restrict the user's focus to a specific range of events within a window. Thus, ripper abstracts widgets and their properties and generates a GUI tree. Each EFG will have an adjacency matrix which helps in identifying the edge between two nodes. If the value is 0, there is no edge between 2 nodes. If the value is 1, there exists an edge. If the value is 2, there exists an edge to itself.

In our implementation, the pictorial representation of the graph is drawn with the help of the adjacency matrix present in the EFG file. The *GraphViz* tool is used for displaying the graph. The EFG file is first converted into dot file format, and then fed into the *GraphViz* which generates the nodes and edges based on the adjacency matrix.

C. Dependency analysis

Dependency analysis is required in order to achieve the concurrency of test automation. The dependent nodes are clustered in order to identify the test cases to be executed in different cycles. To generate a cluster, loops within the graph have to be identified. Hence, the input to this component would be the EFG generated through ripper.

The dependency can be identified w. r. t features, data or function. We are planning to implement a high level dependency based on the connectivity existing within the application based on the EFG. However, this dependency does not provide the complete dependency existing within

the model. Therefore, we are using the Donald B. Johnson algorithm to identify the elementary cycles in a graph at first. This is based on the search for the strongly connected components within the graph. The two important modules of the program are given below. This piece of code identifies the list of objects that contains list of nodes of all elementary cycles in the graph.

D. Test case generation

The application components are nested and hidden once the application is launched. This makes the task of test case generation tedious and time consuming. Thus, the Event flow graph is used to generate the test case for AUT. The test criteria for generating the test cases have been kept that each event in the EFG covers at least 5 test cases. The graph traversal method is used to traverse through the EFG to identify the starting main screen and to track the count of the traversal. The maximum limit for the events in a single event sequence has been kept as 50. The test case inserts the connecting events in the test case to make it executable on the real GUI.

E. Test script runner

Test script runner takes the generated test cases and the model as the input. For each test case, the runner checks the corresponding edges and events in the GUI tree and EFG model for knowing the window and widget information on which the events are executed. The Test runner constitutes of Selenium Grid, Appium nodes, Android Devices and emulators. To begin with the runner process, selenium is first registered as a hub to control all the nodes configured. After successful launch of selenium grid server, appium nodes are registered at different ports. Each appium node has its own configuration.

```

-----
T E S T S
-----
Running com.saucelabs.appium.DependencyTests
Cycle 2 - Test 6
Cycle 2 - Test 7
Cycle 3 - Test 1
Cycle 3 - Test 2
Cycle 1 - Test 1
Cycle 1 - Test 2
Cycle 1 - Test 3
Cycle 1 - Test 4
Cycle 2 - Test 1
Cycle 2 - Test 2
Cycle 2 - Test 3
Cycle 2 - Test 4
Cycle 2 - Test 5
Tests run: 13, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 259.26 sec

```

Figure 4 Test results samples

The node configuration includes capabilities of the appium node. The browser name, version, platform etc can be defined here. The configuration part includes the hubHost, which is the ip address of the machine on which selenium grid is running. The ip address of the hub Host and URL are where the machine appium is running on. The hub Port is 4444 by default, where the selenium grid is listening for requests. Different appium nodes can be registered at different ports by assigning different port numbers.

Based on the data dependencies, test scripts are grouped into cycles. Each cycle has test scripts based on the paths for the dependant node. As shown in Figure 4, test results are

shown after execution of all test cycles. For example, Cycle 2 covers test scripts for one node whose date is dependent on other nodes. Hence, different cycles are color coded.

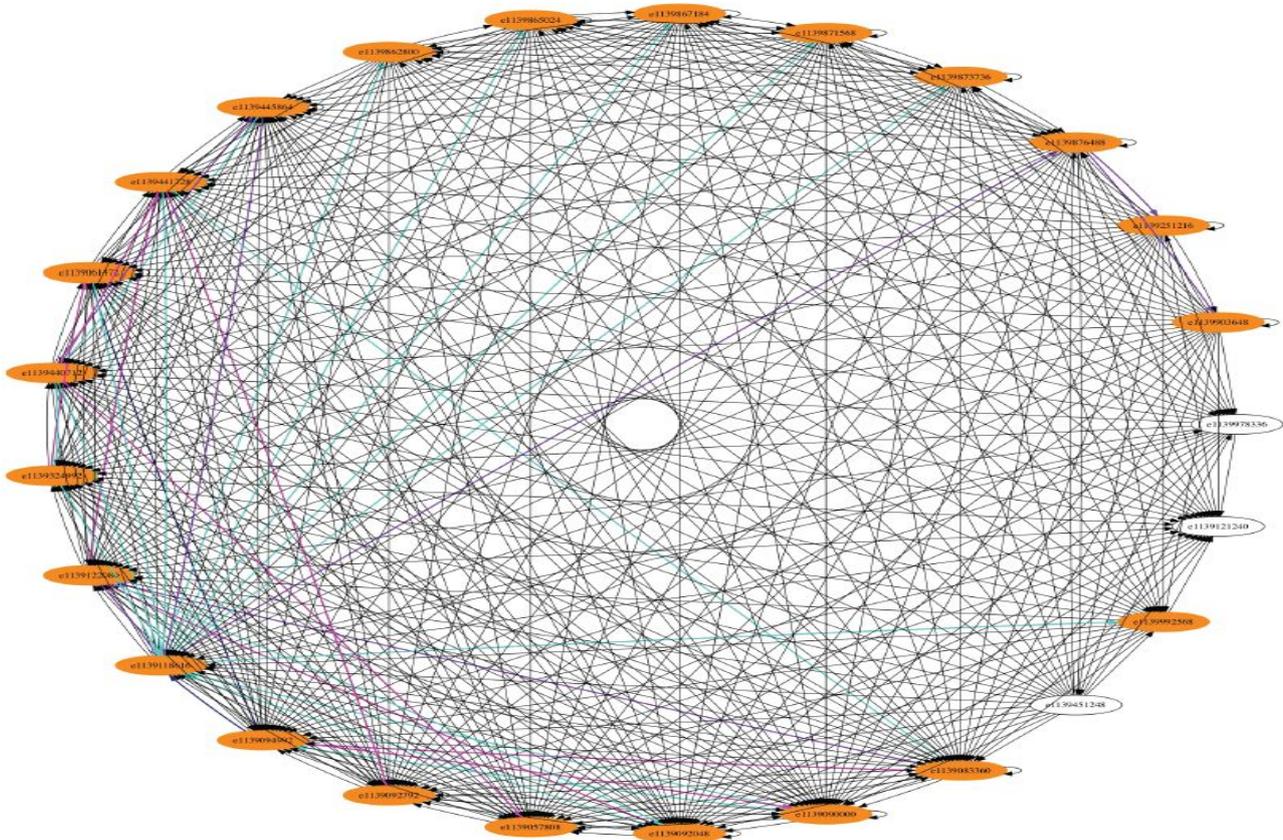


Figure 5 Event Flow Graph with node coverage analysis

As depicted in Figure 5, nodes are reached by finding their xpath. The Figure shows the tests results after execution of all test cycles. Each test-script determines a dependant path. All possible node paths covered are color coded in the graph based on the cycles. For cycle 1 the color code is Magenta, cycle 2 it is Cyan and cycle 3 is Purple. For any test script that fails the color code is red. The tippy tipper app has 25 nodes altogether. We achieved 92% node coverage by covering 22 nodes through our test scripts as depicted in Figure 5. The nodes covered are colored in orange. The uncovered paths are in depicted in black. The color codes are illustrated as follows.

Magenta - Cycle 1 Test scripts – Scenarios covered

Cyan – Cycle 2 Test scripts – Scenarios covered

Purple – Cycle 3 Test scripts – Scenarios covered

Black – Scenarios not covered

Orange – Covered nodes

F. Limitations, and Experience Learned

The proposed approach suffers from some limitations. For example, the syntax is currently dependent on variables.

Thus if the variables associated change, the step definitions need to be modified. As a complex problem in the known space, the new id needs to be extracted from the app properties. Therefore, there is tremendous scope to automate and reduce this dependency on the variables in the future work.

Initially, to set up with the infrastructure, we planned to support test automation where in the hub manager should be robust enough to back and forth data between nodes and test servers/database. Also, since diverse platforms are supported, the same test script written should work on mobile devices running on multiple platforms. Additionally, test scripts are run on multiple devices simultaneously so that scalability is achieved. Furthermore, it has to be taken care that the final solution is fault tolerant, i.e., if a single script fails in one of the devices, it should not fault the other scripts running on the device. Nevertheless, it should not stop the scripts running on other mobile devices. Hence, the automation solution has the challenge of building concurrent, scalable, fault tolerant, unified mobile test automation tool.

VI. Conclusions and Future Work

This paper specified the requirements, design, schedule and budget plan related to our project from the industry perspective. All the team members contributed equally in the entire process and we had great scope for learning from each other. In this paper we aim to develop a tool that performs concurrent test automation for mobile applications. We will make use of open source tools to achieve our aim through this project. The approach aimed at solving several challenges faced by the existing tools in the market. The achieved solution provides GUI test automation of android apps based on model and data dependencies.

Currently our approach supports a unified central test automation platform for only Android OS. It can be further enhanced by adding GUI ripper for iOS apps. This way it can be extended to iOS OS as well. In addition to this, the test runner can be deployed on Amazon EC2 or OpenStack to provide efficient approach for scalability. By then scalability can be achieved at grid level, which is at node level currently.

References

- [1] Y. Huang and L. Lu, "Apply ant colony to event-flow model for graphical user interface test case generation," IET Software, pp.50-60, Feb.2012.
- [2] A.M. Memon and M.B. Cohen, "Automated testing of GUI applications: Models, tools, and controlling flakiness," in 2013 Int.Conf. Software Engineering (ICSE), pp.1479-1480.
- [3] A. Li, Z. Qin, M. Chen and J. Liu, "ADAutomation: An Activity Diagram Based Automated GUI Testing Framework for Smartphone Applications," in 2014 Int. Conf. Software Security and Reliability, pp. 68-77.
- [4] D. Amalfitano, A.R. Fasolino and P. Tramontana, "A GUI Crawling-based technique for Android Mobile Application Testing," in 2011 Int. Conf. Software Testing, Verification and Validation Workshop (ICSTW), pp. 252-261.
- [5] V. Gaur, V. Ragnathan and V. Prakash (2011), "Mobile Test Automation Solutions," Hexaware Technologies., Jamesburg, NJ, Mobile Accelerator White Paper, Available on <http://hexaware.com/casestudies/hs-it-wp-1.pdf>.
- [6] Testing Strategies and Tactics for Mobile Applications, Keynote Systems, Inc., White Paper, Available at <http://www.keynote.com/resources/whitepapers/testing-strategies-tactics-for-mobile-applications>.
- [7] A. Memom, I. Banerjee and A. Nagarajan, "GUI Ripping: Reverse Engineering of Graphical User Interfaces for Testing," in 2003 Working Conf. Reverse Engineering (WCRE), pp. 260-269.
- [8] J. Bo, L.Xiang and G. Xiaopeng, "Mobile Test: A Tool Supporting Automatic Black Box Test for Software on Smart Mobile Devices," in 2007 Int. Workshop Automation of Software Test, pp. 8.
- [9] P. K. Govindasamy(2012), "Selecting the Right Mobile Test Automation Strategy: Challenges and Principles," cognizant 20-20 insights Inc., Available at <http://www.cognizant.com/InsightsWhitepapers/Selecting-the-Right-Mobile-Test-Automation-Strategy-Challenges-and-Principles.pdf>.
- [10] J. Gao, X. Bai, W. T. Tsai, T. Uehara, "Mobile App Testing – A Tutorial", IEEE Computer – Special Issue on Software Validation, Vol 47, No.2, pp.46-55, 2014.
- [11] J. Gao, C.C.Y. Toyoshima, D.K. TLeung, "Engineering on the Internet for Global Software Production", Computer journal, Vol 32, No.5, pp. 38-47, May 1999.
- [12] M. Vieira, J. Leduc, B. Hasling, R. Subramanyan, J. Kazmeier, "Automation of GUI testing using a model-driven approach", International workshop on Automation of software test, pp.9-14, 2006.
- [13] J. Gao, C. Tao, "Modeling mobile application test platform and environment: testing criteria and complexity analysis", Industry Contributions to Test Automation and Model-Based Testing, pp.28-33, 2014.
- [14] A. M. Memon, M. E. Pollack, M. L. Soffa, "Hierarchical GUI Test Case Generation Using Automated Planning", IEEE Transactions on Software Engineering - Special issue on 1999 international conference on software engineering, Vol 27, No.2, February 2001.
- [15] C. Siemens, "The Search for Mobile App Test Automation", blog, November 19, 2013, Available at <http://engineering.zillow.com/the-search-for-mobile-app-testautomation/>
- [16] Test Automation Tools for Mobile Applications: A brief survey, white paper, HSC PROPRIETARY, Available at http://www.hsc.com/Portals/0/Uploads/Articles/hsc_whitepaper_mobileTestAutomation_22Feb2013634971268468845610.pdf
- [17] Sebastian Bauersfeld, A Metaheuristic Approach to Automatic Test Case Generation for GUI-Based Applications, 22 August 2011, available at <http://www2.informatik.huberlin.de/swt/dipl/bauersfeld-2011.pdf>
- [18] G.J. Cong, D. Bader, Lockfree Parallel Algorithms: An experimental Study, available at, <http://www.cc.gatech.edu/~bader/papers/lockfree-HiPC2004.pdf>.
- [19] A. Memom, I. Banerjee, B. N. Nguyen, B. Robbins. The first decade of GUI ripping: Extensions, applications, and broader impacts, in Working Conf. Reverse Engineering (WCRE 2013), pp 11-20.
- [20] S. Anand et al., "Automated Concolic Testing of Smartphone Apps," Proc. ACM SIGSOFT 20th Int'l Symp. Foundations of Software Eng. (FSE12), 2012, pp. 1–11.
- [21] D. Amalfitano et al., "Using GUI Ripping for Automated Testing of Android Applications," Proc. 27th IEEE/ACM Int'l Conf. Automated Software Eng. (ASE 12), 2012, pp. 258–261.
- [22] H. Muccini, A. D. Francesco, and P. Esposito, "Software Testing of Mobile Applications: Challenges and Future Research Directions", Proceedings of International Workshop on Automatic Software Test Automation, 2012, pp 29-35.

Acknowledgement

This paper is supported by the National Natural Science Foundation of China under Grant No.61402229 and No.61502233; the Open Fund of the State Key Laboratory for Novel Software Technology (KFKT2015B10), and the Postdoctoral Fund of Jiangsu Province under Grant No.1401043B.

Redroid: A Regression Test Selection Approach for Android Applications

Quan Do^{*}, Guowei Yang^{*}, Meiru Che[†], Darren Hui[†], Jefferson Ridgeway[§]

^{*} Department of Computer Science, Texas State University, San Marcos, TX 78666, USA

[†] Department of Electrical and Computer Engineering, University of Texas, Austin, TX 78712, USA

[§] Elizabeth City State University, Elizabeth City, NC 27909, USA

Email: ^{*} {q_d2, gyang}@txstate.edu, [†] {meiruche, darren_hui}@utexas.edu, [§] jdridgeway787@students.ecsu.edu

Abstract—As the mobile platform pervades human life, much research in recent years has focused on improving the reliability of mobile applications on this platform, for example by applying automatic testing. However, researchers have primarily considered testing of single version of mobile applications. Although regression testing has been extensively studied for desktop applications, and many efficient and effective approaches have been proposed, these approaches cannot be directly applied to mobile applications. We first present a bug study on real-world Android bugs to show the existence of regression bugs, which motivates the need for an efficient regression test selection technique for Android applications. Next, we introduce Redroid, a new approach to regression test selection for Android applications. Our approach leverages the combination of static impact analysis and dynamic code coverage, and identifies a subset of test cases for re-execution on the modified application version. We implement our approach for Android applications, and demonstrate its efficacy through an extensive empirical study.

I. INTRODUCTION

Mobile devices have become ubiquitous in modern society. The mobile platform is separating itself from a variety of areas of desktop applications such as entertainment, e-commerce and social media. Thus, developers are required to produce high quality mobile applications (or simply, “apps”) in terms of portability, reliability and security. In recent years, a great deal of research has been performed to improve the reliability of mobile apps on mobile platform, for example, by applying automatic testing [5], [15], [6], [19], [8], [16], [7].

Mobile apps are evolving over time, for example, to cope with new requirements or to fix bugs, and regression testing—a process of validating modified software to ensure that changes are correct and do not adversely affect other features of the software—needs to be performed on the new versions of the mobile apps. However, the majority of the research is focused on testing of single version of mobile apps. Although regression testing has been extensively studied for desktop applications, and many efficient and effective approaches have been proposed [10], [18], [13], [17], [11], [12], [20], these approaches cannot be directly applied to mobile apps. A key factor that causes the incompatibility is the difference between the mobile platform’s system architecture and the desktop platform’s. For example, although Android apps are developed using the Java language, they use the Dalvik Virtual Machine

as a runtime environment, which is significantly different from the Java Virtual Machine.

This paper first presents a bug study based on 10 real-world Android apps from Google Code Repository [3]. The study shows that there are regressions for Android apps during evolution, and an efficient and effective regression testing approach is highly needed for this area. This paper then introduces Redroid, a new approach to regression test selection for Android apps. Given a test suite that was performed on the original Android app, and the two versions involved in a change, Redroid identifies a subset of the tests that must be re-executed to test the new Android version. Leveraging the combination of static impact analysis with coverage information that is dynamically generated at runtime, our approach identifies a subset of tests to check the behaviors of the modified version that can potentially be different from the original version. We developed a prototype tool for Redroid, and conducted an evaluation based on two real-world Android apps, which shows that our approach can significantly reduce the number of tests for re-execution after an Android app is modified.

The remainder of this paper is organized as follows: we present a bug study on open source Android apps in Section II. We present our approach in Section III, and then evaluate it in Section IV. We discuss related work and conclude the paper in Sections V and VI.

II. BUG STUDY

We conducted a study to investigate real-world Android bugs with an aim to find how these change-related bugs (regressions) are manifested in Android apps.

We selected Android apps from Google Code Repository [3] based on four specific criteria: 1). large number of downloads, 2). large number of bug reports, 3). long development history, and 4). wide range of apps from different categories. The Google Code Repository houses over 900 apps, and enables users to give feedback on their apps and provide insight to bugs that the app may have while developers are unaware of. The feedback or bug reports that users submit to Google Code Repository for the specific app are given labels and types as that bug is being worked on by the developer(s). Table I lists the 10 apps that were selected for this study. The number

TABLE I
BUG STUDY RESULTS BASED ON REAL-WORLD ANDROID APPS FROM GOOGLE CODE REPOSITORY.

Application	Category	# Downloads	# Bug Reports	Time Span (yrs)	# Regressions
Ankidroid	Education	1,000,000-5,000,000	2,645	6	15
ConnectBot	Communication	1,000,000-5,000,000	688	6	2
Android Wifi Tether	Communication	380,000	1,965	4	5
Ebookdroid	Productivity	1,000,000-5,000,000	937	4	10
Android SMS	Tool	70,000	195	5	4
Android Shuffle	Productivity	50,000-100,000	330	5	6
Android Privacy Guard	Communication	100,000-500,000	166	2	3
Open GPS Tracker	Travel	100,000-500,000	432	4	1
Electric Sleep	Health	100,000-500,000	217	2	3
DroidWall	Tool	1,000,000-5,000,000	318	3	10

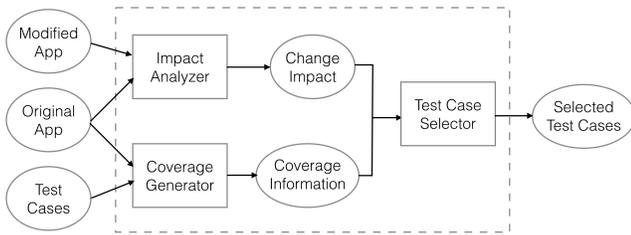


Fig. 1. Redroid Overview.

of downloads and category per app are retrieved from the Google Play store. These 10 apps, chosen from six categories, have 2-6 years of history, and all of them have more than 50k downloads. All of these apps have more than a hundred bug reports, and some of them have over two thousand bug reports.

For each reported bug, there are eight possible labels : Invalid, WontFix, Duplicate, Accepted, Started, Fixed, Fix-Pending, and Verified. The types of each bug report include enhancements and defects. We manually analyzed bug reports with the label “Fixed” and the type “Defect”, which also signifies that the issue has been solved with an updated version of the app and that the app did have a bug. We found that regression bugs do exist in these apps. In particular, 15 regression bugs are found in Ankidroid. Note that our analysis may not find all regression bugs in these apps as we require the bug reports to explicitly mention that the bug is caused by changes, such as changes to the app’s source code, to library (i.e., a version update of the Android operating system), to app’s configuration settings, or to hardware configurations.

We also analyzed these bugs with an aim for bug replication; however, it turns out that replication is difficult since the quality of the report is poor even though there were 10 strong app candidates, due to one or more of the following conditions: insufficient information of replication of the bug and/or how the bug was fixed, the developer(s) not providing the apk file and/or source code files on the Google Code Repository or Github repository, or the source code was not able to be compiled in either the Eclipse Integrated Development Environment (IDE) or Android Studio.

This study shows that there are regressions for Android apps during evolution, and an efficient and effective regression testing approach is highly needed for this area.

III. APPROACH

A. Overview

Given two Android app versions and an initial test suite, Redroid automatically computes the test cases that need to be re-executed on the modified app version. An overview of the approach is shown in Figure 1. There are three main components in the framework: impact analyzer, coverage generator and test case selector.

The impact analyzer takes as input the original app and its the modified version, and generates impacted code that contains changes between the two versions. It uses the algorithm in Dejavu [18] for computing the change impact. Dejavu constructs control-flow graph (CFG) representations of the methods in the two app versions P and P’, in which individual nodes are labeled by their corresponding statements. Following identically-labeled edges, Dejavu performs a simultaneous depth-first graph walk on a pair of CFGs G and G’ for each method and its modified version in P and P’ to find code changes. Given two edges e and e’ in G and G’, if the code associated with nodes reached by e and e’ differs, e is called a dangerous edge as it leads to code that may cause program executions to exhibit different behavior. However, a special handling is needed to find pairs of anonymous inner classes (AICs) in two app versions. Different from regular Java procedures, which are defined as a separated method, AIC is defined as an inner procedure of another procedure.

The coverage generator collects coverage information while the original test suite is executed on the original app version. It computes the lines of code that are executed by each test case. While there are tools such as EMMA [2] that can be used to compute the code coverage during Android testing, we find that EMMA cannot provide the coverage information for each individual test case when a set of test cases are executed in one time. In other words, to get the coverage for each test case, we have to build and run one test case at a time. Therefore, in this work, we automatically instrument each block in bytecode level, so that when executed the instrumented code generates some execution logs, where we can identify the blocks executed by each test.

Test case selector takes as input the change impact information from impact analyzer and the coverage information from coverage generator, and selects for re-

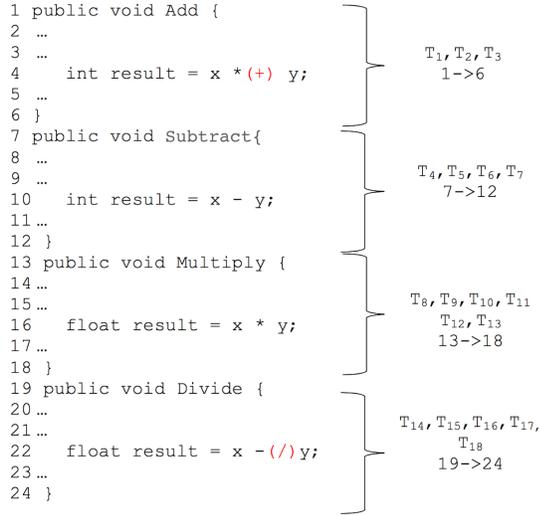


Fig. 2. The four arithmetic operations and their corresponding tests.

execution test cases that are affected by the changes. Since the impact analyzer outputs changed blocks using the same format as used by the coverage generator, the test case selector can easily check whether a test case executes any changed block; if so, that test case is selected for re-execution, as this test case can execute the change and thus its execution on the modified app version may exhibit different behaviors.

B. Illustrating Example

Simple Calculator is a basic calculator that takes as input two numbers and performs four arithmetic operations: addition, subtraction, multiplication and division, which correspond to buttons “+”, “-”, “*”, and “/” respectively. Figure 2 shows part of its implementation for these four operations. 18 test cases are created to test the app’s functionality. By running all the 18 test cases, two bugs are found in the original app, and two fixes are made to lines 4 and 22, where the operators “*” and “-” are replaced by “+” and “/” (in the parentheses), respectively.

After modifying an implementation, the app is required to be re-tested in order to assure the two fixes remove the bugs that are found previously and do not adversely affect other program behaviors. Traditional approach simply re-executes all the test cases in the original test suite on the modified app. However, the modifications may not impact all the test cases, and simply re-running all test cases may not be an efficient way to check the modified app.

Our approach combines change impact analysis and coverage information to reduce the number of test cases for re-execution, and only re-executes the test cases that can potentially reveal different behaviors from the original app version. First, by running all the test cases in the original test suite on the original app, our approach uses coverage generator to collect the coverage information of each test case, i.e., the lines of code that are executed by each test case. As shown in Figure 2, lines 1 → 6 are covered by test cases T_1 , T_2 , and T_3 ; lines 7 → 12 are covered by test cases T_4 ,

TABLE II
ANDROID APPS SELECTED FOR EVALUATION. EACH APP HAS TEST CASES MAINTAINED BY THE DEVELOPERS FOR TESTING.

Apps	Classes	Methods	LOC	Versions	Test Cases
Inetify	63	356	1,500	15	206
AndStatus	250	2,700	15,000	15	99

T_5 , T_6 , and T_7 ; lines 13 → 18 are covered by test cases T_8 , T_9 , T_{10} , T_{11} , T_{12} , and T_{13} ; and lines 19 → 24 are covered by test cases T_{14} , T_{15} , T_{16} , T_{17} , and T_{18} . Moreover, impact analyzer is used to find code changes, and it finds that lines 4 and 22 are changed. Combining this change information and the computed coverage information, our approach selects test cases T_1 , T_2 , T_3 , T_{14} , T_{15} , T_{16} , T_{17} , and T_{18} for re-execution on the modified app, since the execution traces of these test cases contain the changes. It eliminates 10 out of 18 test cases for re-execution, which is a 55.6% reduction.

IV. EVALUATION

Since our technique relies on the output of the coverage generator (CG), we first evaluate the efficiency and preciseness of the CG, by considering two research questions:

- **RQ1:** How efficient is CG compared to the exiting coverage analysis tool EMMA?
- **RQ2:** Does CG achieve the same level of preciseness compared to EMMA?

We then evaluate the overall cost of Redroid relative to the traditional approach which re-runs all test cases (ReTestAll) by considering two research questions:

- **RQ3:** How does the number of test cases selected by Redroid compare to the traditional ReTestAll approach?
- **RQ4:** How does the time cost of applying Redroid compare to the traditional ReTestAll approach?

A. Artifacts

Although 10 apps were studied for our bug study in Section II, no tests were found for these apps. Thus, we selected two other open source Android apps for evaluation: Inetify [4] and AndStatus [1]. For each of the two selected apps, Table II provides information on its associated number of class files (Classes), number of methods (Methods), number of lines of code (LOC), number of basic blocks (Blocks), number of versions we used for evaluation (Versions), and number of test cases. The test cases are found from the projects and maintained by the developers for testing the functionalities of these two apps.

Inetify provides two features related to Wifi networks. First, the app gives a notification if a Wifi network does not provide Internet access. Secondly, it automatically activates Wifi when being near a Wifi network and deactivates Wifi otherwise. AndStatus allows users to login multiple social app accounts such as Twitter and Pump.io. It can combine multiple accounts from all networks into one Timeline and allow users to read and post even if they are offline.

To evaluate our approach, we required multiple versions of each app being analyzed. Because multiple versions of these apps were not available, we generated versions by manually creating mutants of the available app (v_0). When creating mutants, we considered a broad range of changes that can be applied to the code: *change type*, *change location*, and *number of changes*. The change types include modification, deletion, and addition of source code statements. Changes are introduced at different locations: (1) control statements including if statements, case statements, and while, for, and do loops, (2) non-control statements, (3) general locations such as top, middle and bottom of the program source code, (4) Anonymous Inner Class (AIC). AIC procedures are specially considered because they are special Android features which are not existent in regular Java programs, and they are generated as separate class files by the Android Development Environment. Each mutant has one, three, and five changes. For each app, we created five mutants v_1 to v_5 each with one change, five mutants v_6 to v_{10} each with three changes, and five mutants v_{11} to v_{15} each with five changes. Thus, in total we created 15 versions for Inetify and 15 versions for AndStatus, respectively.

B. Variables and Measures

1) *Independent Variables*: The independent variables that we used in the empirical study are the different techniques used for computing coverage and for selecting test cases. To study RQ1 and RQ2, we use our coverage generator (CG), which is based on instrumentation, and the existing coverage tool EMMA [2], a built-in testing framework for Android Platform. To study RQ3 and RQ4, we use Redroid and the traditional ReTestAll approach which re-tests all test cases.

2) *Dependent Variables*: We selected three dependent variables and measures which are ultimately related to the preciseness and cost of coverage generation techniques and test case selection techniques. Given an original app P , which is modified to a new version P' . Note that the costs can be measured differently depending on what technique is being applied. CG and EMMA are only applied to the original app version P . Redroid is applied to both P and P' .

The first dependent variable is *execution time*. To study RQ1, we need to measure execution time to compare the efficiency of CG versus EMMA. To study RQ4, we also need to measure execution time to compare the time cost of Redroid and ReTestAll. The time cost of Redroid is divided to the time for performing static impact analysis, coverage generation, and test selection—which is considered as the overhead of applying Redroid and the time for executing the selected test cases.

The second dependent variable is *code coverage*, which is used to study RQ2. It is measured in terms of the number of executed blocks and the percentage of executed blocks out of the total number of blocks.

The third dependent variable is *the number of selected test cases*. To study RQ3, we need to measure the number of

selected test cases for each test case selection technique, since the goal of our approach is to reduce the number of test cases and thus reduce the cost of performing regression testing. We note that reduction in number of test cases does not necessarily correlate with reduction in time cost, because not all test cases are the same in terms of their execution cost. For example, some test case's execution may cover a large portion of an application and may take long time to execute; while some test case's execution may cover a small portion of the application and may take short time to execute.

C. Experiment Setup

We ran Redroid using the Android Development Environment, together with the Ant tool and Eclipse. The study was performed on a Windows operation system running at 3.4GHz with 8GB of memory. We automated the process from analyzing change impact, generating code coverage report, to selecting test cases by running a batch file under Windows operation system.

For each original app version v_0 , we ran all the test cases in the original test suite with the application of CG and EMMA.

For each mutant app version v_k ($k > 0$), we performed Redroid, which selects test cases by analyzing v_k and v_0 , and runs only the selected test cases on v_k ; we also performed ReTestAll, which re-runs all the test cases in the original test suite on v_k .

D. Results and Analysis

In this section, we present the results of our experiments, and analyze the results with respect to our four research questions.

Table III presents the results of applying the two code coverage analysis tools CG and EMMA. In the table, for each app we list the total number of blocks, the number of executed blocks, the percentage of executed blocks out of the total blocks, and the time cost for coverage analysis, computed by CG and EMMA, respectively.

Table IV presents the results of applying the two test case selection techniques ReTestAll and Redroid on Inetify. In the table, for each mutant version, we list the number of selected test cases and the time cost for running these tests using ReTestAll; we also list the overhead caused, the number of selected test cases and the time cost for executing these test cases using Redroid.

Table V presents the results of applying the two test case selection techniques ReTestAll and Redroid on AndStatus. We list similar types of results as in Table IV.

RQ1: How efficient is CG compared to the exiting coverage analysis tool EMMA?

In Table III, we can see that, for Inetify, EMMA took 341 seconds to get the coverage information for each test case in the test suite, while CG only took 136 seconds, which is more than 60% reduction in time cost. While for AndStatus, the reduction is not that much, CG is still more efficient than EMMA. The main reason for the reduction achieved by CG is because

TABLE III
RESULTS OF CG VS. EMMA

Apps	Number of Test Cases	CG			EMMA				
		Total Blocks	Excuted Blocks	Time (SS)	Total Blocks	Excuted Blocks	Time (SS)		
Inetify	206	7,403	4,930	66.6%	136	6,691	4,452	66.5%	341
AndStatus	99	71,304	41,071	57.6%	497	65,904	37,770	57.3%	595

TABLE IV
TEST CASE SELECTION RESULTS FOR INETIFY

Versions	# Changes	Redroid		
		Overhead (SS)	# Test Cases	Time (SS)
v1	1	4	56	37
v2	1	4	38	25
v3	1	4	39	26
v4	1	4	36	24
v5	1	4	53	35
v6	2	4	108	71
v7	2	4	63	41
v8	2	4	95	62
v9	2	4	111	73
v10	2	4	86	56
v11	3	4	136	96
v12	3	4	140	91
v13	3	4	126	82
v14	3	4	162	106
v15	3	4	171	112

TABLE V
TEST CASE SELECTION RESULTS FOR ANDSTATUS

Versions	# Changes	Redroid		
		Overhead (SS)	# Test Cases	Time (SS)
v1	1	6	8	40
v2	1	6	9	45
v3	1	6	4	20
v4	1	6	10	50
v5	1	6	8	40
v6	2	6	40	200
v7	2	6	37	185
v8	2	6	18	90
v9	2	6	46	230
v10	2	6	44	220
v11	3	6	55	275
v12	3	6	67	335
v13	3	6	64	320
v14	3	6	70	350
v15	3	6	66	330

using EMMA we have to build and run each test at a time to compute its code coverage.

RQ2: Does CG achieve the same level of preciseness compared to EMMA?

In Table III, we can see that CG generated 7,403 blocks while EMMA generated 6,691 blocks, which is a 712-block difference for *Inetify*. Moreover, CG generated 71,304 blocks and EMMA generated 65,904 blocks, which is a 5,400-block difference for *AndStatus*. One possible reason for this difference is that during the process of instrumentation, we insert some virtual blocks such as entry blocks and ending blocks for each CFG, which can lead to more blocks to be counted. Since we do not have knowledge of EMMA's implementation, we are unable to verify what kind of measurement is used by EMMA to count the number of total generated blocks.

Similarly, we can also find the difference in the executed blocks for both *Inetify* and *AndStatus*.

Despite the difference in the number of total blocks and in the number of executed blocks, CG and EMMA achieved almost the same code coverage in terms of percentage. We find a 0.1% difference in code coverage for *Inetify*, and a 0.3% difference in code coverage for *AndStatus*. These differences are so small that they can be neglected. Even if they cannot be neglected, since CG provides more coverage than EMMA, CG can be considered as conservative if not as precise as EMMA, therefore, it is safe to use CG's output for computing test cases.

RQ3: How does the number of test cases selected by Redroid compare to the traditional ReTestAll approach?

From Table IV, we can see that for *Inetify*, while *ReTestAll* selected all the 206 test cases, *Redroid* selected a range from 36 to 172 test cases, achieving 16% to 83% reduction in the number of selected test cases. Moreover, we find the the number of selected test cases varies for mutant versions with the same number of changes, e.g., for mutant versions v_0 to v_5 where only one change is involved in each mutant, *Redroid* selected five different numbers of test cases. This shows that different changes can actually have different impact on behaviors of the app, and thus lead to different selections of test cases. Despite that, unsurprisingly we find that overall the more changes got involved in the mutants, the more test cases were selected, since usually more changes would have more impact. Similar pattern of results can be also found in Table V.

RQ4: How does the time cost of applying Redroid compare to the traditional ReTestAll approach?

From Table IV, we can find that for *Inetify*, there was 4 seconds overhead of applying *Redroid*, due to static impact analysis, coverage generation, and test selection. However, compared the execution time of all the test cases which is 134 seconds, the overhead is not much. Similarly, from Table V we find that for *AndStatus* there was 6 seconds overhead compared to 495 seconds for executing all the test cases.

Moreover, we see that when *Redroid* is performed, due to the reduction of the number of test cases for re-execution, the time cost for executing the selected test is also reduced. Even when the overhead is counted, *Redroid* still achieved reduction in time cost. The biggest time reduction is 79% for version v_4 in *Inetify*, and 95% for version v_3 in *AndStatus*, respectively.

V. RELATED WORK

Android bugs have been studied previously for different purposes. For example, Hu and Neamtiu [14] conducted a bug study to investigate the categories of Android bugs and how Android bugs are manifested; Bhattacharya et al. [9] performed a bug study to understand the bug-fixing process in Android platform and Android-based apps; and Zaeem et al. [21] performed a study to identify user-interaction features for which oracles could be constructed. Different from these studies, our study focuses on investigating bugs that occurred as a result of changes.

Much research has been done to test a single version of Android apps. The behaviors of the tested Android app are explored by using different exploration strategies, including random exploration [5], [15], model-based exploration [6], [19], [8], or systematic exploration [16], [7].

Regression testing has been extensively studied for desktop applications. It is concerned with validating the modified program version after a change. Reusing all of original test suite can be expensive, so various approaches have been developed for re-using tests more effectively by regression test selection [18], [13] and test case prioritization [11], [12], [20].

Regression test selection (RTS) techniques use data on the original and modified program versions, and the original test suite to select a subset of the test suite with which to test the modified program version. One class of RTS techniques, safe techniques (e.g. [18]), guarantee that under certain conditions, test cases not selected could not have exposed faults in the modified program version. Empirical studies have shown that these techniques can be cost-effective. However, due to the difference between desktop platform and Android platform, these approaches cannot be directly applied to mobile apps. Our impact analyzer extends the algorithm in Dejavu [18] to cope with Android bytecode to compute the change impact.

Test case prioritization (TCP) techniques reorder the test cases in the original test suite such that testers can more quickly achieve testing objectives, e.g., to reveal the faults in the modified program [12]. Our work is different as it focuses on regression test selection rather than test case prioritization.

VI. CONCLUSIONS

In this paper, we presented a bug study based on 10 real-world Android apps from Google Code Repository [3], showing that regressions exist for Android apps during evolution; we also presented a manual case study on change impact across the Android activity tree, which shows that regression testing methods can be potentially made more efficient by focusing on changed sections within an Android application. Motivated by these studies, we introduced Redroid, a new approach to regression test selection for Android apps. Given a test suite that was performed on the original Android app, and the two versions involved in a change, Redroid identifies a subset of the tests that must be re-executed to test the new Android version.

Redroid leverages the combination of static impact analysis and coverage information that is dynamically generated at

runtime, and identifies a subset of test cases for re-execution on the modified app version. The execution of those selected test cases can potentially be different from the execution on the original app version. We developed a prototype tool for Redroid, and conducted an evaluation based on two real-world Android apps. Experimental results showed that our approach can significantly reduce the number of tests for re-execution, as well as the time cost for re-executing the selected tests after an Android app is modified.

In future work, we plan to conduct a more comprehensive evaluation of our Redroid technique. We also plan to have a broader range of static analysis that can be used for change impact analysis on Android platform—not only do we focus on changes of the sources code, but also other kinds of changes, such as library changes and hardware changes.

ACKNOWLEDGMENTS

This work is partially supported by the National Science Foundation under Grant No. CNS-1358939.

REFERENCES

- [1] AndStatus. <http://andstatus.org/>.
- [2] EMMA Code Coverage Tool. <http://emma.sourceforge.net/>.
- [3] Google Code Repository. <https://code.google.com/>.
- [4] Inetify. <https://code.google.com/p/inetify/>.
- [5] Monkey. <http://developer.android.com/tools/help/monkey.html>.
- [6] D. Amalfitano, A. R. Fasolino, P. Tramontana, S. De Carmine, and A. M. Memon. Using GUI ripping for automated testing of Android applications. In *ASE 2012*, pages 258–261, 2012.
- [7] S. Anand, M. Naik, M. J. Harrold, and H. Yang. Automated concolic testing of smartphone apps. In *FSE '12*, pages 59:1–59:11.
- [8] T. Azim and I. Neamtiu. Targeted and depth-first exploration for systematic testing of Android apps. In *OOPSLA '13*, pages 641–660.
- [9] P. Bhattacharya, L. Ulanova, I. Neamtiu, and S. C. Koduru. An empirical analysis of bug reports and bug fixing in open source Android apps. In *CSMR '13*, pages 133–143.
- [10] D. Binkley. Semantics guided regression test cost reduction. *IEEE Trans. Softw. Eng.*, 23(8):498–516, Aug. 1997.
- [11] H. Do and G. Rothermel. On the use of mutation faults in empirical assessments of test case prioritization techniques. *IEEE Trans. Softw. Eng.*, 32(9):733–752, Sept. 2006.
- [12] S. Elbaum, A. Malishevsky, and G. Rothermel. Incorporating varying test costs and fault severities into test case prioritization. In *ICSE '01*, pages 329–338.
- [13] M. J. Harrold, J. A. Jones, T. Li, D. Liang, A. Orso, M. Pennings, S. Sinha, S. A. Spoon, and A. Gujarathi. Regression test selection for Java software. In *OOPSLA '01*, pages 312–326.
- [14] C. Hu and I. Neamtiu. Automating GUI testing for Android applications. In *AST '11*, pages 77–83.
- [15] A. Machiry, R. Tahiliani, and M. Naik. Dynodroid: An input generation system for Android apps. In *ESEC/FSE 2013*, pages 224–234, 2013.
- [16] R. Mahmood, N. Mirzaei, and S. Malek. EvoDroid: Segmented evolutionary testing of Android apps. In *FSE 2014*, pages 599–609.
- [17] A. Orso, N. Shi, and M. J. Harrold. Scaling regression testing to large software systems. In *SIGSOFT '04/FSE-12*, pages 241–251.
- [18] G. Rothermel and M. J. Harrold. A safe, efficient regression test selection technique. *ACM Trans. Softw. Eng. Methodol.*, 6(2):173–210, Apr. 1997.
- [19] W. Yang, M. R. Prasad, and T. Xie. A grey-box approach for automated GUI-model generation of mobile applications. In *FASE '13*, pages 250–265.
- [20] S. Yoo, M. Harman, P. Tonella, and A. Susi. Clustering test cases to achieve effective and scalable prioritisation incorporating expert knowledge. In *ISSTA '09*, pages 201–212.
- [21] R. N. Zaeem, M. R. Prasad, and S. Khurshid. Automated generation of oracles for testing user-interaction features of mobile apps. In *ICST '14*, pages 183–192.

Conformance Testing of Balana: An Open Source Implementation of the XACML3.0 Standard

Sung-Ju Fan Chiang

Department of Computer Science
Boise State University
Boise, ID 83725, USA
sungjufanchiang@u.boisestate.edu

Daniel W. Chen

Department of Economics
University of Chicago
Chicago, IL 60637, USA
danielwchen@uchicago.edu

Dianxiang Xu

Department of Computer Science
Boise State University
Boise, ID 83725, USA
dianxiangxu@boisestate.edu

Abstract—As a new generation access control method, Attribute-Based Access Control (ABAC) has gained increasing attention. Currently, Balana is the only open-source implementations of XACML 3.0, which is an OASIS standard for specifying ABAC. Considering that XACML is much more complex than traditional access control models, conformance testing of any XACML implementation is an important problem. Using a non-conformance implementation may lead to misunderstanding of access decisions or even security violations. This paper presents an approach to conformance testing of Balana, focusing on the main elements of the XACML3.0 language, such as targets, rules, policies, and policy sets. In particular, we have thoroughly tested the key rule combining algorithms in policies and policy combining algorithms in policy sets. This has revealed several conformance issues.

Keywords—attribute-based access control; Balana; conformance testing; decision tables; XACML.

I. INTRODUCTION

Access control is a fundamental mechanism for preventing malicious or accidental security violation. An access control policy specifies the conditions under which access to resources can be granted and to whom [12]. With the increasing system complexity, access control methods have evolved from Mandatory Access Control (MAC), Discretionary Access Control (DAC), Role-Based Access Control (RBAC) to Attribute-Based Access Control (ABAC). ABAC combines various attributes of authorization elements into access control decisions [6]. These attributes are predefined characteristics of subjects (e.g., job title and age), resources (e.g., data, programs, and networks), actions, and environments (e.g., current time and IP address).

XACML (eXtensible Access Control Markup Language) [11] is an OASIS standard for specifying ABAC policies in the XML format. It can be used within a large enterprise or across multiple organizations. Currently Balana [1] is the only open source implementations of XACML 3.0. It is worth pointing out that the original open source implementation of XACML from Sun Microsystems, Inc. only supports 1.0 and 2.0. While it is believed to be upgraded to 3.0 in Oracle's Identity Server, the upgraded version is no longer open source. XACML3.0 is much more complex than traditional access control methods such as RBAC. For example, XACML 3.0 provides various combining algorithms to support rule and policy composition. A combining algorithm aims at rendering a single access decision by combining the decisions of individual access

control rules or policies. The standard specification of XACML3.0 lacks a rigorous representation of the semantics of the combining algorithms. Our prior work on the formalization of the semantic differences between various combining algorithms in XACML 3.0 shows that, for any pair of rule (or policy) combining algorithms studied, they can be functionally equivalent with respect to certain rules (or policies) [13]. The similarities and differences among the combining algorithms are subtle. This increases the likelihood of having errors. Thus, conformance testing of any XACML3.0 implementation is an important issue. Using a non-conformance XACML implementation may lead to misunderstanding of access decisions or even security violations.

In this paper, we present our work on the systematic testing of conformance between Balana and XACML3.0. It focuses on the main language elements of XACML3.0, such as targets, rules, policies (including rule combining algorithms), and policy sets (including policy combining algorithms). As XACML is a logic-based language, we use decision tables as the main technique for formulating the semantics of these language elements and their test requirements. Although decision tables are a traditional technique, the particular decision tables resulted from this research provide an accurate understanding of the meanings of the main XACML 3.0 elements. They offer important guidelines for XACML3.0 practitioners. In addition, they are useful for testing other implementations of XACML3.0 in order to verify functional conformance. Based on the decision tables, the conformance tests for targets, rules, and policies are created manually. For policy sets and policy combining algorithms, however, all conformance tests are generated automatically from the existing conformance tests. In our experiment, all conformance tests are executed automatically. The results have revealed several conformance issues in Balana.

The remainder of this paper is organized as follows. Section II gives a brief introduction to the main XACML language elements. Section III describes our conformance testing method. Section IV presents the results of our conformance testing experiment. Section V reviews related work. Section VI concludes this paper.

II. XACML LANGUAGE ELEMENTS

The first-class entities in XACML are policy and policy set. A policy is composed of an optional policy target, one or more rules, and a rule combining algorithm. A policy set consists of

an optional policy set target, one or more sub-policy sets or policies, and a policy combining algorithm. Figure 1 shows the main elements of XACML 3.0 and their relationships.

A rule consists of a target, a condition, and an effect. The target is a logical expression that specifies the set of requests to which the rule is intended to apply. The logical operators are *AnyOf* and *AllOf*. Specifically, a target consists of zero or more *AnyOf* clauses, and each *AllOf* clause is made up of one or more match predicate. The condition is a Boolean expression that refines the applicability of the rule established by the target. Predicates in target and condition are defined over attributes and attribute values (e.g., gender is male).

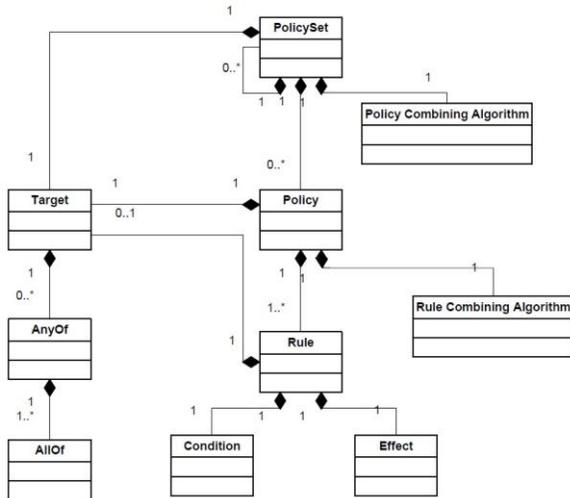


Figure 1. Main language elements of XACML 3.0 [11]

An access request consists of attribute and value pairs. Given a request, the decision of a policy depends on the policy target, the decisions of individual rules in the policy, and the rule combining algorithm. Each rule may yield one of the following decisions if the policy target evaluates to true:

- *Permit*: access is granted when the rule effect is *Permit* and the rule target evaluates to *Match* and the rule condition evaluates to true.
- *Deny*: access is denied when the rule effect is *Deny*, the rule target evaluates to *Match*, and the rule condition evaluates to true.
- *Not-Applicable*, denoted as *N/A* in this paper: either the rule target evaluates to *No Match* or the rule condition evaluates to false.
- *Indeterminate Deny*, denoted as *I(D)*: An error occurred when the rule target or the rule condition was evaluated. The decision could have evaluated to *Deny* if no error had occurred.
- *Indeterminate Permit*, denoted as *I(P)*: An error occurred when the rule target or the rule condition was evaluated. The decision could have evaluated to *Permit* if no error had occurred.

The rule combining algorithm combines the decisions of individual rules into a single policy-level decision. In addition

to the above decisions, a policy decision can be *Indeterminate Deny Permit*, denoted as *I(DP)*. *I(D)*, *I(P)* and *I(DP)* will be a plain *Indeterminate* if it is the final decision returned by the XACML engine.

XACML3.0 provides 11 rule combining algorithms. Four of them are for compatibility support of old versions - *Legacy Ordered-deny-overrides*, *Legacy Permit-overrides*, *Legacy Ordered-permit-overrides*, and *Legacy Ordered-permit-overrides*. In Balana, the implementations of *Ordered-deny-overrides* and *Ordered-permit-overrides* are the same as *Deny-overrides* and *Permit-overrides*. Thus this paper focuses on the following five rule combining algorithms:

- *Deny-overrides*: Intended for those cases where a *Deny* decision should have priority over a *Permit* decision;
- *Permit-overrides*: Intended for the cases where a *Permit* decision should have priority over a *Deny* decision.
- *Deny-unless-permit*: Intended for those cases where a *Permit* decision should have priority over a *Deny* decision, and an *Indeterminate* or *N/A* must never be the result.
- *Permit-unless-deny*: Intended for those cases where a *Deny* decision should have priority over a *Permit* decision, and an *Indeterminate* or *N/A* must never be the result.
- *First-applicable*: Rules are evaluated in the order in which they are listed. If a rule's target matches and condition evaluates to *True*, then return the rule's effect (*Permit* or *Deny*). If the target or condition evaluates to *False*, the next rule is evaluated. If no further rule exists, then return *N/A*. If an error occurs, then return *Indeterminate*, with the appropriate error status.

Given a request, a policy set yields one of the six decisions: *Permit*, *Deny*, *N/A*, *I(D)*, *I(P)*, and *I(DP)*. It depends on the policy set target, the decisions of individual policy sets and policies in the policy set, and the policy combining algorithm. XACML 3.0 specifies 12 policy-combining algorithms. Similar to the reasons for the selected rule combining algorithms, this paper focuses on the six policy combining algorithms: *Deny-overrides*, *Deny-unless-permit*, *Permit-overrides*, *Permit-unless-deny*, *First-applicable*, and *Only-one-applicable*.

III. THE CONFORMANCE TESTING METHOD

Although XACML has a number of language elements, policy and policy set are the first-class executable units. Thus, an executable conformance test case must include a policy (or policy set), an access request, and expected response. The policy (or policy set) and access request are called test input, whereas the expected response is called test oracle. Test oracles of all conformance tests are defined per the XACML3.0 standard specification. When creating conformance tests for other language elements such as policy targets, rule targets, rule conditions, and rules, we also need to create policy files. When a conformance test is executed with Balana, we verify whether the actual response produced by Balana is the same as the test oracle. If not, Balana does not conform to the standard specification and the test is called a non-conformance test.

The main technique used for the conformance testing is decision tables because XACML is essentially a logic-based language. Unlike the traditional mathematical logic that has two truth values (true and false), XACML is more like multi-valued logic due to the consideration of error conditions. For example, a match predicate could evaluate to *True*, *False*, or *Indeterminate* (which means an error has occurred during the evaluation).

We build decision tables for the main XACML language elements based on their semantics described in the standard specification. The decision tables capture the requirements of conformance testing. Concrete conformance tests are then created or generated to cover every entry. Specifically, the tests for targets, rules, and policies are created manually, whereas the tests for policy sets (i.e., sample policy sets and access requests) are generated from the existing tests for policies (i.e., sample policies and access requests).

In the following, we present the test requirements of the main XACML language elements in the form of decision tables. The decision tables are not only useful for the conformance testing, but can help XACML users get an accurate understanding of XACML policies. We start with the basic elements (i.e., targets and rules) and then focus on policies and policy sets.

A. Conformance Testing of Targets

The target in a rule, policy, or policy set consists of zero or more *AnyOf* clauses. An *AnyOf* clause consists of a sequence of *AllOf* clauses. An *AllOf* clause consists of a sequence of match predicates, which are the basic element of targets. A match predicate matches an attribute name with an attribute value. A match predicate evaluates to *True*, *False*, or *Indeterminate* (i.e., an error has occurred during evaluation). A target can evaluate to *Match*, *No Match*, or *Indeterminate*. Table 1 shows the decision table of target evaluation per the XACML3.0 standard specification. *AllOf* is similar to the logical operator “and” – an *AllOf* clause evaluates to *Match* if and only if all match predicates in the *AllOf* clause evaluate to true. It evaluates to indeterminate if one of the match predicate evaluates to *Indeterminate*. *AnyOf* is similar to the logical operator “or”. An *AnyOf* clause evaluates to true if one of the *AllOf* clauses evaluate to true.

Table 1 essentially specifies the minimum test requirements for the target element in XACML 3.0. Our test design ensures that each entry in the decision table is covered by at least one conformance test.

TABLE 1. DECISION TABLE FOR TARGET EVALUATION

AnyOf1				AnyOf2				Decision
AllOf1		AllOf2		AllOf3		AllOf4		
Mat ch1	Mat ch2	Mat ch3	Mat ch4	Mat ch5	Mat ch6	Mat ch7	Mat ch8	
T	T	T	T	T	T	T	T	M
T	T	T	T	T	T	T	I	M
T	T	T	T	T	T	T	F	M
T	T	T	T	T	I	T	I	I
T	T	T	T	T	I	T	F	I
T	T	T	T	T	F	T	F	N
T	T	T	I	T	T	T	I	M

T	T	T	I	T	T	T	F	M
T	T	T	I	T	I	T	I	I
T	T	T	I	T	I	T	F	I
T	T	T	I	T	F	T	F	N
T	T	T	F	T	T	T	F	M
T	T	T	F	T	I	T	I	I
T	T	T	F	T	I	T	F	I
T	T	T	F	T	F	T	F	N
T	I	T	I	T	I	T	I	I
T	I	T	I	T	I	T	F	I
T	I	T	I	T	F	T	F	N
T	I	T	F	T	I	T	F	I
T	I	T	F	T	F	T	F	N
T	F	T	F	T	F	T	F	N

T stand for "True", I stand for "Indeterminate", F stand for "False", M stand for "Match", N stand for "No match".

B. Conformance Testing of Rules

A rule consists of rule target, rule condition, and rule effect (either *Permit* or *Deny*). Rule target (or rule condition) is optional and evaluates to *Match* (or *True*) when it is absent. Rule condition evaluates to *True*, *False*, or *Indeterminate*. Table 2 shows the decision table of rules, where *D/C* refers to “don’t care”. In the case of rule condition, *D/C* means that the evaluation result of the rule condition is either *True*, *False*, or *Indeterminate*. In the case of rule effect, *D/C* means either *Permit* or *Deny*. For example, when the rule target evaluates to *Indeterminate* and the rule effect is *Permit*, the rule’s decision is *I(P)*, regardless of the evaluation result of the rule condition. When the rule target evaluates to *No Match*, the rule’s decision is *N/A* regardless of the rule condition and rule effect. Our test design ensures that each entry is covered by at least one test.

TABLE 2. DECISION TABLE FOR RULE EVALUATION

Evaluation of Rule Target	Evaluation of Rule Condition	Rule Effect	Rule Decision
Match	True	Permit	Permit
Match	True	Deny	Deny
Match	False	D/C	N/A
Match	Indeterminate	Permit	I(P)
Match	Indeterminate	Deny	I(D)
No Match	D/C	D/C	N/A
Indeterminate	D/C	Permit	I(P)
Indeterminate	D/C	Deny	I(D)

C. Conformance Testing of Policies

The main elements of a policy include a policy target, one or more rules, and a rule combining algorithm. Table 3 shows a general decision table about how a policy is evaluated per the XACML3.0 specification. Given a request, if it matches the policy target, then the policy decision depends on the decisions of individual rules and the rule combining algorithm. If the request does not match the policy target, the policy decision is *N/A* regardless of the decisions of individual rules. If the policy target evaluates to *Indeterminate* (i.e., an error has occurred), the policy decision depends on the rule decisions.

The decision tables for rule combining algorithms are created according to the descriptions and pseudo code in the XACML3.0 specification. They are applied when a given

request matches the policy target. Table 4 shows the decision table for the *Deny-overrides* rule combining algorithm. The decision of an individual rule can be *Permit*, *Deny*, *N/A*, *I(D)*, or *I(P)*. Table 4 shows all of the 25 combinations of two rules. In particular, if one rule evaluates to *Deny*, the combined decision is *Deny* - this reflects the meaning of *Deny-overrides*. When there are more than two rules, the combined decision of $n-1$ rules can be combined with the n -th rule to obtain the policy-level decision.

TABLE 3. DECISION TABLE FOR POLICY EVALUATION

Evaluation Result of Policy Target	Rule Decisions	Policy Decision
Match	Deny	Specified by the rule-combining algorithm
Match	Permit	
Match	N/A	
Match	I(D)	
Match	I(P)	
No Match	D/C	N/A
Indeterminate	Deny	I(D)
Indeterminate	Permit	I(P)
Indeterminate	N/A	N/A
Indeterminate	I(D)	I(D)
Indeterminate	I(P)	I(P)

TABLE 4. DECISION TABLE FOR THE DENY-OVERRIDES RULE-COMBINING ALGORITHM

Deny-overrides		Decision of the first rule				
		Permit	Deny	N/A	I(D)	I(P)
Decision of the second rule	Permit	Permit	Deny	Permit	I(D)	Permit
	Deny	Deny	Deny	Deny	Deny	Deny
	N/A	Permit	Deny	N/A	I(D)	I(P)
	I(D)	I(D)	Deny	I(D)	I(D)	I(DP)
	I(P)	Permit	Deny	I(P)	I(DP)	I(P)

When the name of a rule combining algorithm is also used as a policy combining algorithm, the decision table for the policy combining algorithm (e.g., Table 6) is more general than the decision table of the corresponding rule combining algorithms (e.g., Table 5). Thus, this paper does not present the decision tables of other rule combining algorithms.

Based on the decision tables for the policy evaluation and the rule combining algorithms, we create policy files and request files to cover all entries of each decision table. A policy file and a request file form the input of a conformance test. The test oracle is the corresponding policy decision in the decision table. For an entry of *D/C*, we create a conformance test to cover each possible value of that entry.

To execute the tests automatically, we specify all the conformance tests (policy file, request file, and oracle value) in a spreadsheet. For each entry of the spreadsheet, our test execution framework will invoke Balana with the corresponding policy file and request file, compare the actual response from Balana with the oracle value, and report the verdict (pass/fail).

D. Automated Conformance Testing of Policy Sets

The main elements for a policy set include a policy set target, sub-policies or policy sets, and a policy combining

algorithm. The evaluation of a policy set is similar to that of policy evaluation. Table 5 shows the general decision table for policy set evaluation. Given a request, if it matches the policy set target, then the policy set decision depends on the decisions of individual sub-policies/policy sets, and the policy combining algorithm. If the request does not match the policy set target, the policy set decision is *N/A* regardless of the decisions of individual sub-policies/policy sets. If the policy set target evaluates to Indeterminate (i.e., an error has occurred), the policy set decision depends on the decisions of individual sub-policies/policy sets.

TABLE 5. DECISION TABLE FOR POLICY SET EVALUATION

Policy Set Target	Decisions of Sub-Policies or Policy Sets	Policy Set Decision
Match	Deny	Specified by the policy-combining algorithm
Match	Permit	
Match	N/A	
Match	I(D)	
Match	I(P)	
No Match	D/C	N/A
Indeterminate	Deny	I(D)
Indeterminate	Permit	I(P)
Indeterminate	N/A	N/A
Indeterminate	I(D)	I(D)
Indeterminate	I(P)	I(P)
Indeterminate	I(DP)	I(DP)

Tables 6-11 are the decision tables for the six policy combining algorithms. An important feature of this work is that we automatically generate conformance tests for policy sets from the decision tables of the policy combining algorithms and the existing conformance tests for policies.

Let us use the *Deny-overrides* policy combining algorithm as an example. Table 6 is its decision table. We need to create a conformance test for each entry in Table 6 (i.e., a total of 36 tests for Table 6). Consider the entry where the decision of the first policy in the policy set is *Permit* and the decision of the second policy in the policy set is *Deny*. Our goal is to create a policy set file and a request file such that (a) the policy set has two policies, (b) the policy combining algorithm is *Deny-overrides*, (c) the first policy evaluates to *Permit* with respect to the request, and (d) the second policy evaluates to *Deny* with respect to the request. We generate such a policy set file and a request file as follows:

- (1) Find an existing policy test including a policy file and a request file such that the policy decision should be *Permit*. Let us denote the policy file as *P1* and the request file as *R1*.
- (2) Find an existing policy test including a policy file and a request file such that the policy decision should be *Deny*. Let us denote the policy as *P2* and the request as *R2*.
- (3) Find the attribute names in both policy tests. For any attribute that appears in both policy tests, rename the attribute in the second policy test (*P2* and *R2*). Let *P2'* and *R2'* be the revised policy and request.

- (4) Generate a policy set file from $P1$ and $P2'$ using the *Deny-overrides* as the policy combining algorithm. The policy set target is set to empty (which always evaluates to *Match*) or move the target of $P1$ or $P2'$ to the policy set target.
- (5) Generate a request file by composing the attributes and their values in $R1$ and $R2'$.
- (6) The oracle value of the policy set conformance test is *Deny*, according to the decision table.

Note that the renaming in step (3) is critical. It resolves the naming conflicts – the same attribute from different tests may have different meanings. Without this step, (4) and (5) would not guarantee that the first policy in the policy set evaluates to *Permit* or the second policy in the policy set to *Deny*.

For each policy combining algorithm, the generated conformance tests (including policy set file, request file, and oracle value) are specified in a spreadsheet. For each entry of the spreadsheet, the test execution framework will call Balana with the corresponding policy set file and request file, compare the actual response from Balana with the oracle value, and report the verdict (pass/fail).

TABLE 6. DECISION TABLE FOR THE DENY-OVERRIDE POLICY-COMBINING ALGORITHM

Deny-overrides		<i>Decision of the first policy or policy set</i>					
		Permit	Deny	N/A	I (D)	I (P)	I (DP)
<i>Decision of the second policy or policy set</i>	Permit	Permit	Deny	Permit	I (DP)	Permit	I (DP)
	Deny	Deny	Deny	Deny	Deny	Deny	Deny
	N/A	Permit	Deny	N/A	I (D)	I (P)	I (DP)
	I (D)	I (DP)	Deny	I (D)	I (D)	I (DP)	I (DP)
	I (P)	Permit	Deny	I (P)	I (DP)	I (P)	I (DP)
	I (DP)	I (DP)	Deny	I (DP)	I (DP)	I (DP)	I (DP)

TABLE 7. DECISION TABLE FOR THE PERMIT-OVERRIDE POLICY-COMBINING ALGORITHMS

Permit-overrides		<i>Decision of the first policy or policy set</i>					
		Permit	Deny	N/A	I (D)	I (P)	I (DP)
<i>Decision of the second policy or policy set</i>	Permit	Permit	Permit	Permit	Permit	Permit	Permit
	Deny	Permit	Deny	Deny	Deny	I (P)	I (DP)
	N/A	Permit	Deny	N/A	I (D)	I (P)	I (DP)
	I (D)	Permit	Deny	I (D)	I (D)	I (DP)	I (DP)
	I (P)	Permit	I (P)	I (P)	I (DP)	I (P)	I (DP)
	I (DP)	Permit	I (DP)	I (DP)	I (DP)	I (DP)	I (DP)

TABLE 8. DECISION TABLE FOR THE DENY-UNLESS-PERMIT POLICY-COMBINING ALGORITHMS

Deny-unless-permit		<i>Decision of the first policy or policy set</i>					
		Permit	Deny	N/A	I (D)	I (P)	I (DP)
<i>Decision of the second policy or policy set</i>	Permit	Permit	Permit	Permit	Permit	Permit	Permit
	Deny	Permit	Deny	Deny	Deny	Deny	Deny
	N/A	Permit	Deny	Deny	Deny	Deny	Deny
	I (D)	Permit	Deny	Deny	Deny	Deny	Deny
	I (P)	Permit	Deny	Deny	Deny	Deny	Deny
	I (DP)	Permit	Deny	Deny	Deny	Deny	Deny

TABLE 9. DECISION TABLE FOR THE PERMIT-UNLESS-DENY POLICY-COMBINING ALGORITHMS

Permit-unless-deny		<i>Decision of the first policy or policy set</i>					
		Permit	Deny	N/A	I (D)	I (P)	I (DP)

<i>Decision of the second policy or policy set</i>	Permit	Permit	Deny	Permit	Permit	Permit	Permit
	Deny	Deny	Deny	Deny	Deny	Deny	Deny
	N/A	Permit	Deny	Permit	Permit	Permit	Permit
	I (D)	Permit	Deny	Permit	Permit	Permit	Permit
	I (P)	Permit	Deny	Permit	Permit	Permit	Permit
	I (DP)	Permit	Deny	Permit	Permit	Permit	Permit

TABLE 10. DECISION TABLE FOR THE FIRST-APPLICABLE POLICY-COMBINING ALGORITHMS

		<i>Decision of the first policy or policy set</i>					
		Permit	Deny	N/A	I (D)	I (P)	I (DP)
<i>Decision of the second policy or policy set</i>	Permit	Permit	Deny	Permit	I (D)	I (P)	I (DP)
	Deny	Permit	Deny	Deny	I (D)	I (P)	I (DP)
	N/A	Permit	Deny	N/A	I (D)	I (P)	I (DP)
	I (D)	Permit	Deny	I (D)	I (D)	I (P)	I (DP)
	I (P)	Permit	Deny	I (P)	I (D)	I (P)	I (DP)
	I (DP)	Permit	Deny	I (DP)	I (D)	I (P)	I (DP)

TABLE 11. DECISION TABLE FOR THE ONLY-ONE-APPLICABLE POLICY-COMBINING ALGORITHM

		<i>Decision of the first policy or policy set</i>			
		Permit	Deny	N/A	I (Indeterminate)
<i>Decision of the second policy or policy set</i>	Permit	I	I	Permit	I
	Deny	I	I	Deny	I
	N/A	Permit	Deny	N/A	I
	I	I	I	I	I

IV. RESULTS OF CONFORMANCE TESTING

Our conformance testing has revealed several non-conformance cases as summarized below. It is worth pointing out that these cases do not necessarily lead to security violations in an XACML application. It depends on how the responses are handled by the application's policy enforcement point. However, understanding the differences between Balana and the XACML standard specification is important for the users of Balana to correctly enforce access control policies.

The current implementation of the *Permit-overrides* rule and policy combining algorithm does not conform to the XACML3.0 specification with respect to the error conditions. Table 12 shows the three non-conformance tests for which Balana's responses are different from the test oracles per the XACML standard specification (refer to the decision table for *Permit-overrides* in Table 7). When the decisions of two policies in a policy set are *N/A* and *Indeterminate Deny*, or both *Indeterminate Deny*, the decision of the policy set should be *Indeterminate Deny*. However, the actual response of Balana is *N/A*.

TABLE 12. NON-CONFORMANCE TESTS OF THE PERMIT-OVERRIDES POLICY-COMBINING ALGORITHM

Non-Conformance Test	Test Input		Test Oracle per XACML	Actual Result by Balana
	<i>Decision of Policy 1</i>	<i>Decision of Policy 2</i>		
1	N/A	I(D)	I(D)	N/A
2	I(D)	N/A	I(D)	N/A
3	I(D)	I(D)	I(D)	N/A

The current implementation of the *Deny-overrides* combining algorithm also has four non-conformance tests as

shown in Table 13. For example, when the decisions of two policies in a policy set are $I(D)$ and $Permit$, the decision of the policy set should be $I(DP)$. However, the actual response of Balana is $I(D)$. When the decision of this policy set is the final response to the user, there will be no difference because both $I(DP)$ and $I(D)$ will result in a plain *Indeterminate*. However, when such a policy set is used by other policy sets, the non-conformance results may lead to different final decisions for access requests.

TABLE 13. NON-CONFORMANCE TESTS OF THE DENY-OVERRIDES POLICY-COMBINING ALGORITHM

Non-Conformance Test	Test Input		Test Oracle per XACML	Actual Result by Balana
	Decision of Policy 1	Decision of Policy 2		
1	$I(D)$	Permit	$I(DP)$	$I(D)$
2	$I(D)$	$I(P)$	$I(DP)$	$I(D)$
3	Permit	$I(D)$	$I(DP)$	$I(D)$
4	$I(P)$	$I(D)$	$I(DP)$	$I(D)$

In addition, the initial version of Balana used in our project failed the conformance tests of the *Permit-unless-deny* policy-combining algorithm. Examination of the source code revealed that the bugs resulted from the copy-paste of the *Deny-unless-permit* policy-combining algorithm. This has been fixed in the current version of Balana, though.

V. RELATED WORK

Existing work on XACML-related testing has focused on the testing of XACML policies, not the implementation of the XACML standard. Thus, no literature is directly comparable to this paper. In Cirg [8], tests are generated from counterexamples produced by the change-impact analysis of two synthesized versions of an XACML policy. The difference of the two versions of a policy targets a test coverage goal (e.g., rule, or condition). Targen [9] is a test generator for XACML policies that derives access requests to satisfy all the possible combinations of truth values of the attribute id-value pairs found in a given policy. Access requests generated by Cirg and Targen typically use a limited number of subject, resource, action, and environment attributes. A real request, however, could use any combination of attributes. Because requests are encoded in XML, they must conform to the XML Context Schema. To address this issue, Bertolino et al., have developed several test generation algorithms [2][3][4][5]. These algorithms can generate requests that use more than one subject, resource, action, or environment attribute. They can also produce robustness tests, where invalid attribute values are generated randomly. Li et al. have applied symbolic execution technique to generation of access requests for testing XACML policies [7]. They convert the policy under test into semantically equivalent C Code Representation (CCR) and symbolically execute CCR to create test inputs and translate the test inputs to access control requests. Xu et al., have proposed a fault-based testing approach for determining existence or absence of incorrect combining algorithms in XACML 3.0 policies [12].

VI. CONCLUSIONS

We have presented an approach to the conformance testing of Balana, which is currently the only open source implementation of the XACML3.0 standard. Our experiment has revealed subtle conformance issues. The decision tables used to define the conformance test requirements are not only useful for testing XACML3.0 implementations, but also provide important guidelines for understanding the meanings of XACML3.0 language elements. In particular, the various rule combining algorithms and policy combining algorithms have subtle differences and similarities.

ACKNOWLEDGMENT

This work was supported in part by US National Science Foundation (NSF) under grants CNS 1359590 and 1461133. Dr. Yunpeng Zhang and Mr. Ning Shen participated in the initial conformance testing of XACML combining algorithms.

REFERENCES

- [1] Balana, "Open source XACML 3.0 implementation," <http://xacmlinfo.org/2012/08/16/balana-the-open-source-xacml-3-0-implementation/>, 2012.
- [2] A. Bertolino, S. Daoudagh, F. Lonetti, and E. Marchetti. "Automatic XACML requests generation for policy testing." Fifth IEEE International Conference on Software Testing, Verification and Validation (ICST), 2012, pp.842-849.
- [3] A. Bertolino, S. Daoudagh, F. Lonetti, and E. Marchetti. "The X-CREATE Framework-A Comparison of XACML Policy Testing Strategies." *Proc. of the 8th International Conference on Web Information Systems and Technologies (WEBIST)*. pp.155-160.
- [4] A. Bertolino, S. Daoudagh, F. Lonetti, and E. Marchetti. "Xacmut: Xacml 2.0 mutants generator." Sixth IEEE Int'l Conf. on Software Testing, Verification & Validation Workshops (ICSTW). 2013, pp.28-33.
- [5] A. Bertolino, S. Daoudagh, F. Lonetti, E. Marchetti and L. Schilders. "Automated testing of extensible access control markup language-based access control systems." *Software, IET 7.4* (2013), pp.203-212.
- [6] V. C. Hu, D. Ferraiolo, R. Kuhn, A. Schnitzer, K. Sandlin, R. Miller and K. Scarfone. "Guide to Attribute Based Access Control (ABAC) Definition and Considerations." NIST Special Pub 800 (2014): 162.
- [7] Y. C. Li, Y. Li, L. Z. Wang, and G. Chen. "Automatic XACML Requests Generation for Testing Access Control Policies." *Proc. of the 26th International Conf. on Software Engineering and Knowledge Engineering (SEKE'14)*, Vancouver. July 2014.
- [8] E. Martin and T. Xie. "Automated test generation for access control policies," in *Supplemental Proc. of ISSRE*, November 2006.
- [9] E. Martin, and T. Xie. "Automated test generation for access control policies via change-impact analysis." *Proceedings of the Third International Workshop on Software Engineering for Secure Systems*. IEEE Computer Society, 2007, pp.5-11.
- [10] E. Martin, and T. Xie. "A fault model and mutation testing of access control policies." *Proceedings of the 16th International Conference on World Wide Web*. ACM, 2007, pp.667-676.
- [11] OASIS, "eXtensible Access Control Markup Language (XACML) Version 3.0," <http://www.oasisopen.org/committees/xacml/>, 2013.
- [12] D. Xu, N. Shen, Y. Zhang, "Fault-based testing of combining algorithms in XACML 3.0 policies," *Proc. of the 27th Int'l Conf. on Software Engineering and Knowledge Engineering (SEKE'15)*, 2015.
- [13] D. Xu, Y. Zhang, N. Shen, "Formalizing semantic differences between combining algorithms in XACML 3.0 policies," *Proc. of the 2015 International Conference on Software Quality, Reliability and Security (QRS'15)*, pp. 163-172. Vancouver, Canada. August 2015.

QCC:A novel cluster algorithm based on Quasi-Cluster Centers

Jinlong Huang¹, Qingsheng Zhu¹, Lijun Yang¹, Dongdong Cheng¹

¹Chongqing Key Lab. of Software Theory and Technology, College of Computer Science,
Chongqing University, Chongqing 400044, China
Email: qs Zhu@cqu.edu.cn; 352720950@qq.com

Abstract—Cluster analysis is aimed at classifying elements into categories on the basis of their similarity. And cluster analysis has been widely used in many areas such as pattern recognition, and image processing. In this paper, we propose an approach based on the idea that the density of cluster centers are highest in its k nearest neighborhood or reverse k nearest neighborhood, and clusters is divided by sparse region. We firstly define the similarity between clusters. Based on this idea, no matter non-spherical data or complex manifold data, the proposed algorithm is applicable. And the proposed algorithm has a certain capacity on outliers detection. We demonstrate the power of the proposed algorithm on several test cases. Its clustering performance is better than DBSCAN, DP and K-AP clustering algorithms.

Keywords—Cluster; Center; Similarity; Neighbor; manifold

I. INTRODUCTION

Clustering is a primary method of data mining and data analysis. The aim of clustering is to classify elements into categories, or clusters, on the basis of their similarity. Clusters are collections of objects whose intra-class similarity is high and inter-class similarity is low. Now the study on clustering algorithm has been very active. Several different clustering methods have been proposed[1]. They can be roughly divided into Partitioning Methods[2-4], Hierarchical clustering[5-7], Density-Based Clustering[8-9], Grid-Based Clustering[10-11], Model-Based Method[12-13].

For many clustering algorithm, it is important that finding cluster center to clustering. In K-means[2] and K-medoids[3] methods, the data was classified to a cluster by a small distance to the cluster center. An objective function, typically the sum of the distance to a set of putative cluster centers, is optimized until the best cluster centers candidates are found. In 2007, Brendan and Delbert proposed a new clustering algorithm by passing messages between data points, called “affinity propagation”(AP)[14]. AP takes as input measures of similarity between pairs of data points. Real-valued messages are exchanged between data points until a high-quality set of exemplars and corresponding clusters gradually emerges. However, AP clustering algorithm can not directly specify the final class number. In order to generate K clusters, Zhang et al. proposed K-AP clustering algorithm[15].

However, these above center-based methods are not able to detect non-spherical clusters[16], since data points are always

assigned to the nearest center. In 2014, Rodriguez et al. proposed a new clustering algorithm in Science, called DP[17]. The DP algorithm has its basis in the assumptions that cluster centers are surrounded by neighbors with lower local density and that they are at a relatively large distance from any points with a higher local density. And the non-spherical shape clusters can be easily detected by DP[17] clustering algorithm. But DP is not application to complex manifold data sets. In 2014, Hongjie et al. proposed the DAAP clustering algorithm[18] that can solve the complex manifold problem by computing the particular similarity that defined in paper[18]. However, the time complexity of DAAP is higher than AP, K-AP and DP for computing the particular similarity that the sum of the Edge-Weight in shortest path. And the clustering result of DAAP is effected by many parameters such as the number of neighbors and clusters, damping coefficient, the maximum iteration. Moreover, generally, the clustering effect of DAAP is bad on datasets with noise points. Detailed description see paper[18].

In this paper, we propose a new clustering method. The proposed algorithm has its basis in the assumptions that the density of cluster centers is the maximum of its neighbors or reverse neighbors and clusters are divided by sparse area. Similar to the above methods, it based on the cluster centers. However, unlike these above methods, the cluster centers of the proposed algorithm are not ‘real’ cluster centers that one center corresponds to one final cluster, but the Quasi-Cluster Center that one Quasi-Cluster Center corresponds to one initial cluster. Then we obtain the final cluster by merging the clusters that the similarity is greater than alpha. The proposed clustering algorithm and it’s related definitions will be detailed described in section 3.

II. RELATED WORK

Most of density based clustering algorithms, such as DBSCAN, DP, AP and K-AP, define the number of neighbors that distance is smaller than d_c as the density of each point. And the density of points as the follows formula.

$$\rho_i = \sum_j^n X(d_{ij} - d_c) \quad (1)$$

Where $X(d_{ij} - d_c) = 1$ if $d_{ij} - d_c < 0$ and $X(d_{ij} - d_c) = 0$ otherwise, and d_c is a cutoff distance. Basically, ρ_i is equal to the number of points that are closer than d_c to point

i. However, once the intra-class density variations is great, the value of d_c is hard to set. For example, as shown in Figure 1, if the value of d_c is set inappropriately, then there are no neighbors in the neighborhood of point a and b within d_c , but the neighbors of the center of C_1 include all of the points of C_1 . Moreover, although point a and b are normal point, they will be regard as the noise in density-based algorithms. And point a and b will be regard as the cluster center in DP algorithm, since the local density of point a and b is the biggest in its neighborhood.

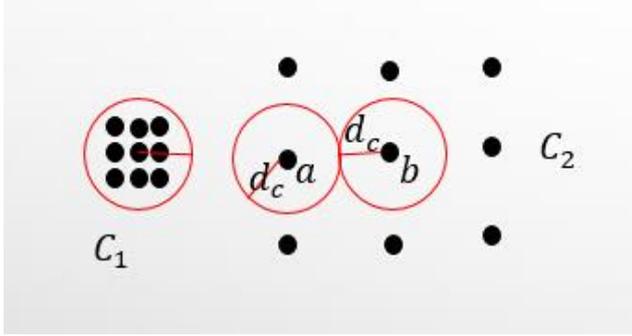


Figure 1. the density variations

As the most of exist density based clustering algorithm, the proposed method need to compute the density of every point so that we can get the cluster centers that the density peaks. In order to avoid the above question, we introduce the follow definitions.

Let D be a database, p and q be some objects in D , and k be a positive integer. We use $d(p,q)$ to denote the Euclidean distance between objects p and q .

Definition1 (K-distance and Density): The k -distance of p , denoted as $K_{dist}(p)$, is the distance $d(p,o)$ between p and o in D , such that:

- (1) For at least K objects $o' \in D/\{p\}$ is holds that $d(p,o') \leq d(p,o)$, and
- (2) For at most $(K-1)$ objects $o' \in D/\{p\}$ it holds that $d(o,o') < d(p,o)$

The $K_{dist}(p)$ can represent the density of the object p . The smaller $K_{dist}(p)$ is, the much denser the area around p is. So, like paper [19], we define the density of p , denoted as $Den(p)$, as the follows equation:

$$Den(p) = 1/K_{dist}(p) \quad (2)$$

Definition2 (K Nearest Neighbor and Reverse K Nearest Neighbor) If $d(p,q) \leq K_{dist}(p)$, then call the object q as the K Nearest Neighbor of p . All of the K Nearest Neighbor compose the K Nearest Neighborhood, denote as $KNN(p)$. Conversely, call the object p as the Reverse K Nearest Neighbor of q , and all of the Reverse K Nearest Neighbor compose the Reverse K Nearest Neighborhood, denote as $RKNN(q)$. The formulation of $KNN(p)$ and $RKNN(p)$ as following:

$$KNN(p) = \{q | d(p,q) \leq K_{dist}(p)\}$$

$$RKNN(q) = \{p | d(p,q) \leq K_{dist}(p)\}$$

III. THE PROPOSED ALGORITHM

In this paper we divide the neighbors of every point into Dense Neighbors and Sparse Neighbor, defined as the **Definition3**.

Definition3 (Dense and Sparse Neighbor): If the density of q is greater than p and $q \in KNN(p)$, then call the object q as the Dense Neighbor of p , denote as $DN(p)$. On the contrary, if the density of q is smaller than p and $q \in KNN(p)$, then q is called as the Sparse Neighbor, denote as $SN(p)$.

Definition4 (Exemplar) If the density of q is the maximum in the neighbors of p and $p \neq q$, then call the object q is the Exemplar of p .

From the Definition of Exemplar, we can know that each point of dataset possess at most one Exemplar. If the density of p is greater than the density of all k nearest neighbors or reverse k nearest neighbors of p , then p is the Exemplar of itself. And we call p is the **Quasi-Cluster Center(QCC)**.

Definition5 (Quasi-Cluster Center) If object p satisfied one of the follows two conditions, then we call p as QCC.

- (1) $\forall q \in KNN(p), Den(p) \geq Den(q)$ or
- (2) $\forall q \in RKNN(p), Den(p) \geq Den(q)$

Figure 2 is the Exemplar Graph(EG) which can be comprised by connecting each point p to its Exemplar. As shown in Figure 2, the parameter $k=30$, points c_1, c_2, \dots, c_7 etc. is QCC that marked in red. Other red points will be treated as outliers that will be explained in **Algorithm1**. And through many experiments and analysis, we find that the number of **Quasi-Cluster Center** appears to get smaller as the parameter k becomes bigger.

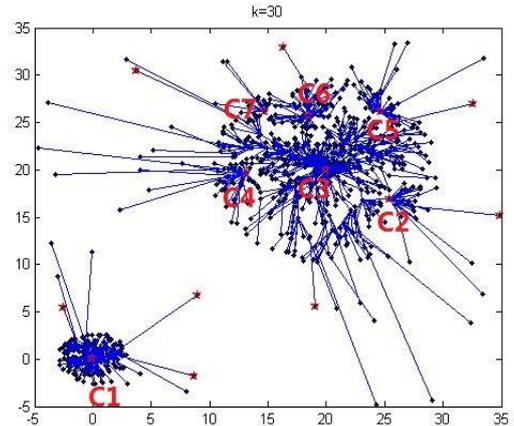


Figure2: Exemplar Graph and Ouasi-Cluster Center

Definition6 (Similarity between clusters): Similarity between clusters C_i and C_j , denote as $Sim(C_i, C_j)$, is defined as the ratios of the number of objects q that $q \in C_i \cap q \in C_j$ and K . The formulation of similarity between clusters as follows:

$$Sim(C_i, C_j) = Num(\{q | q \in C_i \cap q \in C_j\}) / K \quad (3)$$

As shown in Figure 3, We consider the ratios of the number of these points that between two adjacent marked in red and K as the similarity of two adjacent initial clusters. If these two adjacent initial clusters are divided by sparse area, then the similarity of these two cluster is small. In other words, these two clusters are two individual cluster. On the contrary, if these two adjacent initial clusters are connected by density area, the similarity of these two adjacent clusters will be great. And these two clusters will be merged to one cluster. In this way, even if one big cluster was divided to many small clusters because the value of k is small, like Figure1 shows, these small clusters C1, C2, ..., C7 will be merged into one cluster finally.

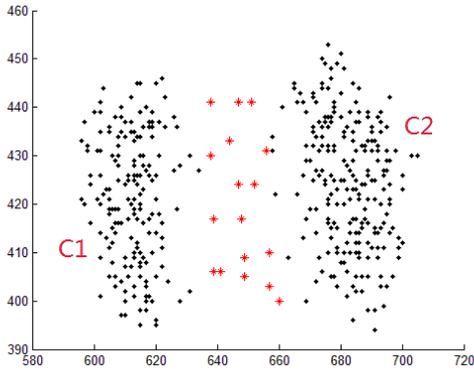


Figure3: The similarity between C1 and C2

Based on the above definitions, we proposed a novel clustering algorithm, named QCC, with the capacity that outlier detecting. The procedure of QCC algorithm is minutely described in **Algorithm1**.

Firstly, the proposed clustering algorithm QCC use the **KNN-Searching** to obtain the KNN and RKNN of each point of dataset D, so we need the parameter k that the number of the neighbors of every points. Then compute the density of each point. Secondly, step 5 in **Algorithm1**, QCC find the Exemplar of each point using **Definition4**, and obtain all of the Quasi-Cluster Center using **Definition5**. After that, QCC obtain the initial clusters.

- (1) QCC arbitrarily find a Quasi-Cluster Center, and classify it and it's Sparse Neighbors to the same cluster C_i .
- (2) Then QCC arbitrarily find a point p in this cluster and classify the Sparse Neighbors of p to cluster C_i , until all points of this cluster have been visited.
- (3) Then QCC find an other Quasi-Cluster Center and repeat the above steps, until all Quasi-Cluster Center have been visited.

In this way, the clusters extended from dense area to the sparse area. As shown in Figure(3), the points marked in red are classified to C1 and C2 at the same time. Then QCC merge all the clusters that similarity greater than alpha into one cluster. If the similarity of cluster C1 and C2 is smaller than alpha, then the red points will classified into the cluster that it's Exemplar belongs to. alpha is a artificial parameter. Generally, the value

Algorithm1: QCC(D,K,alpha) //alpha is the similarity between clusters.

Output: $C=\{c_1, c_2, \dots, c_M\}$

- (1) Initializing Variables: $r=0, K_dis(i)=0, Den(i)=0, KNN(i) = \emptyset, RKNN(i) = \emptyset, SN(i)=\emptyset, Exemplar(i)=i, Sim(c_i, c_j) = 0, Q_{CC} = \emptyset;$
- (2) $[KNN(i), RKNN(i)]=$ **KNN-Searching(D,K)**
//use the KNN-Searching algorithm to obtain the KNN and RKNN of every point.
- (3) For $\forall x \in D$ // compute the K-distance and the Density of the objects in D.
 - a. Find y that the K-th nearest neighbor of x
 - b. $K_dis(x)=||x - y||_2;$
 - c. $Den(x)=1/K_dis(x);$
- (4) For $\forall x \in D$ find the SN(x).
- (5) For $\forall x \in D$ // find the Exemplar of x and obtain the Quasi-Cluster Center.
 - a. $y=\max(Den(KNN(x)));$
 - b. if $y \neq x$ then $Exemplar(x) = y;$
 - c. if $y=x$ then $r=r+1$ and $Q_{CC}(r)=x;$
 - d. $z=\max(Den(RKNN(x)));$
 - e. if $x=z$ then $r=r+1$ and $Q_{CC}(r)=x;$
- (6) For $i=1$ to r //obtain the initial cluster.
 - a. $c_i = \{Q_{CC}(i)\} \cup SN(Q_{CC}(i));$
 - b. For $\forall x \in c_i$
 - c. If $visited(x) \neq true$ then $visited(x)=true$ and $c_i = c_i \cup SN(x);$
- (7) Compute the similarity matrix $Sim(c_i, c_j)$ between the clusters.
- (8) While $\max(Sim(c_i, c_j)) \geq \alpha$
//merge the initial cluster
 - a. $(i,j)=\max(Sim(c_i, c_j));$
 - b. Merge cluster c_i and $c_j;$
 - c. Update the similarity matrix;
- (9) If $\exists (0 < Sim(c_i, c_j) < \alpha)$
 - a. If $x \in c_i \& x \in c_j$ then x is classified to the cluster that it's exemplar belongs to.
- (10)For $i=1:length(C)$
 - a. If number of $c_i < k;$
 - b. Then $\forall x \in c_i$ x is marked as outlier and delete c_i from C;
- (11)Output the final clusters
 $C=\{c_1, c_2, \dots, c_M\}$

of alpha is 0.2 to 0.5. The higher the alpha, more clusters can be obtained.

After the above steps, QCC regard the clusters that the number of points smaller than k as outlier clusters. In other words, the points in these clusters is marked as outliers. So the red points in Figure2 will be regarded as outliers except $C1, C2, \dots, C7$. So QCC will obtain the accurate clustering results as long as the value of k is smaller than the number of points in smallest cluster of dataset. Finally, QCC output the ultimate clusters. So QCC not only cluster the dataset, but also has certain ability of outlier detection.

IV. EXPERIMENTAL ANALYSIS

A. Cluster on Artificial Data Set

We chose four challenging artificial data sets. Data1, taken from [8], consist of two spherical data and two manifold data that one is simple, another is complex, and a few outliers, a total of 582 points. Data2, taken from [20], composed of three spherical data, one complex manifold data and some noise points, a total of 1400 points. Data3, taken from [21], composed of six high density manifold data and some noise points, a total of 8000 points. Data4, taken from [22], composed of one dense spherical cluster and one sparse manifold cluster, a total of 159 points.

In all results, we don't show the decision graph, decide the number of the clusters, of DP. We decide the right number of clusters to Data1, Data3 and Data4. For Data2, we show the best cluster result in repeated test. For DAAP, we set the density factor is assigned as $\rho=2$, the maximum iteration $\text{maxits}=1000$, convergence of iteration coefficient $\text{convits}=100$.

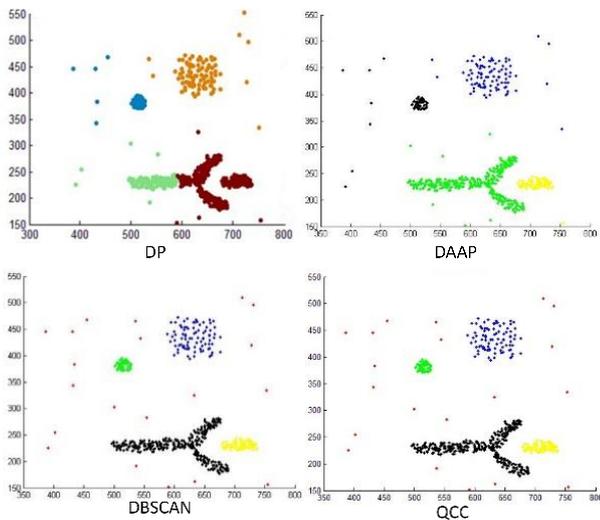


Figure 4: The cluster results of DP, DAAP, DBSCAN and QCC algorithm on Data1.

Figure 4 shows the DP, DAAP, DBSCAN and QCC algorithm's clustering results on Data1. For DAAP, the value of the number(k) of neighbors that used for construct the adjacency matrix is set 6, the value damping coefficient(λ) is 0.9. From this figure, we can see that, as analyses in section1, DP algorithm can correctly cluster the spherical data and simple manifolds data, but can't correctly cluster the complex manifolds data. Data1 is correctly clustered by DAAP, DBSCAN($\text{eps}=15$, $\text{minpoints}=5$) and QCC($k=20$, $\alpha=0.3$). Moreover, DBSCAN

and QCC algorithm detect out the noise points in Data1, but DAAP can't.

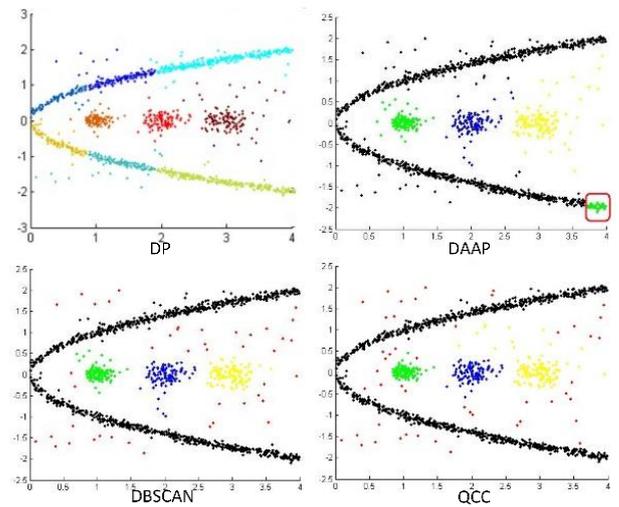


Figure 5: The cluster results of DP, DAAP, DBSCAN and QCC algorithm on Data2.

Figure 5 shows the four algorithm's clustering results on Data2. For DAAP, the value of the number(k) of neighbors that used for construct the adjacency matrix is set 6, the value damping coefficient(λ) is 0.9. DP failed to cluster the complex manifolds data that grouped into 6 clusters. Owing to particular similarity that the sum of the weight of the edge in shortest path, DAAP has some capacity of cluster to complex manifold datasets. However, DAAP failed to cluster the manifolds cluster in Data2. Since the shortest path is too long, so the end region(marked by red square) of the manifold cluster is classified the wrong cluster. Data2 is correctly clustered by DBSCAN($\text{eps}=0.2$, $\text{minpoints}=40$) and QCC($k=20$, $\alpha=0.3$) algorithm. And most of the noise points in Data2 is detected out by DBSCAN and QCC.

Figure 6 shows the four algorithm's clustering results on Data3. Although DP obtain the right number of clusters in Data3 by artificially select the cluster centers in decision graph, three clusters are wrongly clustered among these clusters. DAAP obtain the right number of clusters, but some clusters are wrongly clustered too. Moreover, DAAP mistakenly regard a part of noise as a small normal cluster. The cluster result of DBSCAN is obviously superior to DP and DAAP, and DBSCAN detect out the noise points in Data3. However, some points in normal clusters are treated as noise points. Although QCC($k=80$, $\alpha=6$) failed to detect out the noise in Data3, QCC obtain the right number of cluster without the number of clusters that artificial set, and correctly cluster the normal points.

Figure 7 shows the four algorithm's clustering results on Data4. Same as the results on Data4, although DP and DAAP obtained the right number of clusters by artificially select or set. For the density variations of the two clusters in Data4 is great, DBSCAN($\text{eps}=2$, $\text{minpoints}=5$) failed to correctly cluster Data4. DBSCAN don't obtain the right number of clusters in Data4, and mistakenly treat some normal points as noise. The performance

of QCC(k=5, alpha=0.2) is obviously superior to DP, DAAP and DBSCAN on Data4.

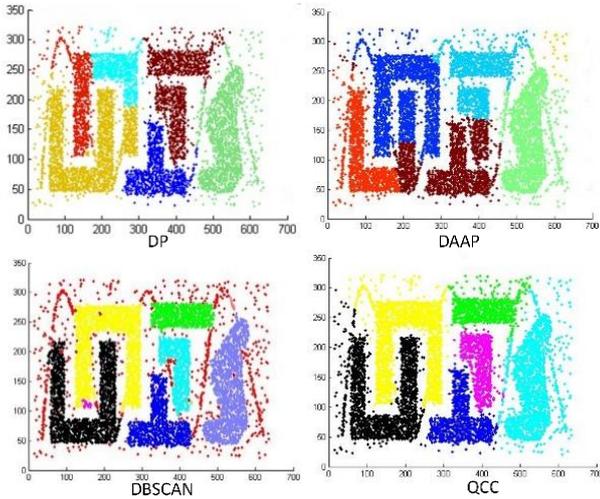


Figure 6: The cluster results of DP, DAAP, DBSCAN and QCC algorithm on Data3.

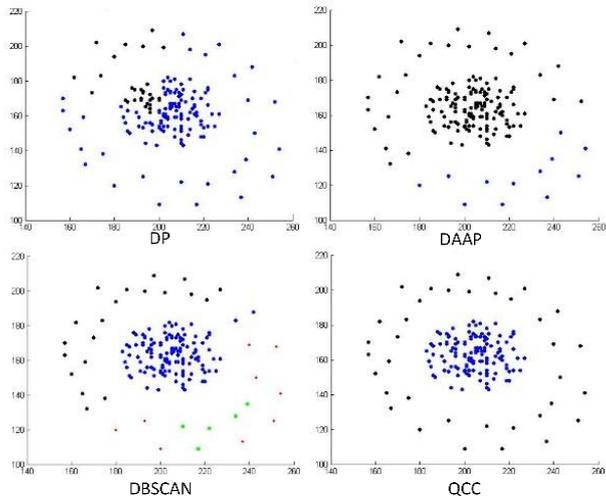


Figure 7: The cluster results of DP, DAAP, DBSCAN and QCC algorithm on Data4.

From the above results and analysis, we can see that, DP algorithm has a certain capacity to cluster non-spherical

data(correctly cluster Data2). But, as shown in the above results, DP algorithm can hardly correctly cluster the complex manifold datasets. DAAP has a certain capacity of cluster to complex manifold datasets. However, DAAP failed to cluster those datasets that include long manifolds data(Data2), lots of noise(Data3) or great density variations clusters(Data4). Although the performance of DBSCAN is superior to DP and DAAP, DBSCAN failed to correctly cluster on Data3 and Data4. So, from the results of artificial datasets, we can see that QCC that proposed in this paper can get the right number of final clusters without human intervention. And the scope of QCC's application is wider than other cluster algorithms. No matter complex manifold datasets or density variations is great, QCC can get satisfactory clustering results.

It should be noted that the value of k should smaller than the number of points of minimum cluster, and the value of alpha is between 0.2 and 0.5 for most data sets. When the bulk density is high, the value of alpha should be greater accordingly, such as Data4. In order to demonstrate the effectiveness of QCC, we also experiment on real datasets as the follows section.

B. Cluster on Olivetti Face Database

Like the paper[17], we also applied the QCC algorithm to the Olivetti Face Database[23], a widespread benchmark for machine learning algorithms, with the aim of identifying, without any previous training, the number of subjects in the database. For this experiment, we used 10 clusters of Olivetti face database. And each cluster is composed of 10 face picture. The size of each picture is <112x92 nint8>. The similarity between two images, denote as S(A,B), was computed by the follows equation.

$$S(A, B) = \frac{\sum_m \sum_n (A_{mn} - \bar{A})(B_{mn} - \bar{B})}{\sqrt{(\sum_m \sum_n (A_{mn} - \bar{A})^2)(\sum_m \sum_n (B_{mn} - \bar{B})^2)}} \quad (4)$$

Here A and B are the subjects of Olivetti Face Database. A_{mn} and B_{mn} represent the pixels of the two subjects picture. The value of S is scaled between 0 and 1. Bigger the value of S is, more similar the two picture is. So we define the distance of two picture, denote as d(A,B), as following equation.

$$d(A, B) = 1 - S(A, B) \quad (5)$$

The density is estimated as **Definition1**.

The results is shown in Figure 8. In the results, faces with same color belong to the same cluster.

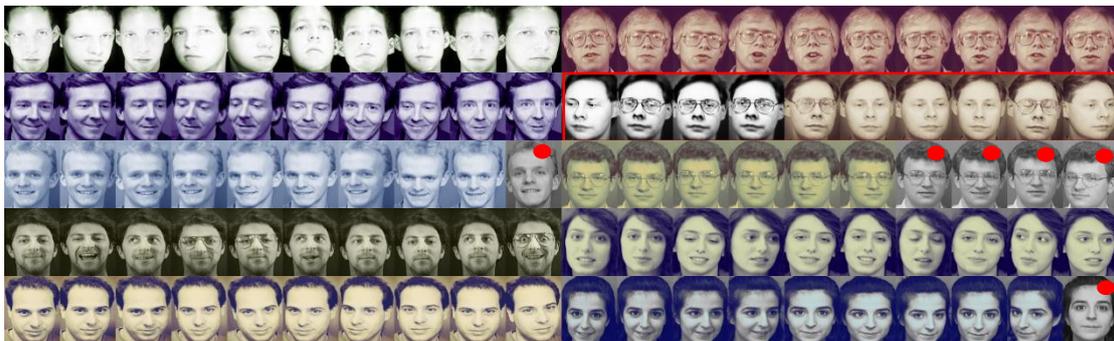


Figure 8: QCC(k=4, alpha=0.2) algorithm's clustering results on Olivetti

In order to intuitively describe the efficiency of QCC, we use two criteria (Purity, Recall) to evaluate the clustering performance. And the calculation formula is as follows:

$$\text{Purity} = \frac{\sum_{i=1}^k \left(\max_{tc \in TC} (tc \cap c_i / N_{c_i}) \right)}{k} \quad (6)$$

$$\text{Recall} = \frac{1}{N} \sum_i^k N_{c_i} \quad (7)$$

Here, Let D be a database and contains Tk clusters TC = {tc₁, tc₂, ..., tc_{Tk}}. The result of clustering algorithm is C = {c₁, c₂, ..., c_k}. N_{c_i} is the number of points of c_i. N is the number of points of whole dataset. The value of Purity and Recall is [0,1], the larger the value of ACC, means the better the clustering performance of the algorithm.

TABLE I. ACC AND RECALL OF THE FOUR ALGORITHM

	DP	DAAP	DBSCAN	QCC
Purity	0.88	0.61	0.98	1
Recall	1	1	0.64	0.94

The QCC algorithm's clustering results on Olivetti Face Database is shown in Figure 8. The results show that Olivetti Face Database was grouped into 11 clusters, because one of the real 10 cluster that within the red border was divided into two clusters. And 6 images that marked with red spot was considered as outliers by QCC algorithm. However, among the 11 clusters, 6 clusters is really correct, 2 clusters is identified 9 face images, one cluster is identified 6 faces in all 10 faces. Moreover, all of the 11 clusters remain pure, namely include only images of the same cluster. So the Purity of QCC is 1 (the best one). The value of Purity of DP and DAAP is 0.88 and 0.61. Although the value of Purity of DBSCAN is 0.98 that closest to QCC, the value of Recall of DBSCAN is the lowest (0.64). Moreover, the Recall of QCC is 0.94 that close to 1.

Through above analysis to results that cluster on artificial data and Olivetti Face Database, it is obvious that the results of QCC outperform the DP, DAAP and DBSCAN algorithm. And we can get the conclusion that QCC algorithm has a more broad application than AP and DP algorithm. QCC algorithm has a certain ability that outliers detecting and can cluster on complex manifold data sets. Furthermore, QCC is not likely to omit any cluster center as DP. So QCC cluster algorithm superior to AP, DP and DBSCAN algorithm.

V. CONCLUSION

In this study, we propose a new cluster algorithm (QCC). The core idea of QCC is that clusters is divided by sparse region. Based on this idea, we define the Quasi-Cluster Centers. Remarkably, the real cluster centers must be included by Quasi-Cluster Centers. So QCC is not likely to omit any clusters. Then QCC obtain the initial clusters by step 5 of Algorithm 1. After this, we define the Similarity between initial clusters. Therefore, QCC applies to complex manifold data sets. Through the experiments on the four artificial datasets, we confirmed that the proposed cluster algorithm (QCC) can correctly cluster on complex manifold data sets that DP, DAAP and DBSCAN can't. The results from the Olivetti Face Database also demonstrated

that QCC is more effective than DP, DAAP and DBSCAN. Furthermore the scope of application of QCC is more extensive.

ACKNOWLEDGMENT

This research was supported by the National Nature Science Foundation of China (No. 61272194).

REFERENCES

- [1] Xu, R. and D. Wunsch, Survey of clustering algorithms. *Neural Networks, IEEE Transactions on*, 2005. 16(3): p. 645-678.
- [2] Han, J. and M. Kamber, *Data mining: concepts and techniques*, Morgan Kaufmann San Francisco, Calif, USA, 2001.
- [3] Kaufman, L. and P.J. Rousseeuw, *Finding groups in data: an introduction to cluster analysis*: John Wiley & Sons. Vol. 344. 2009.
- [4] Ng, R.T. and J. Han, Clarans: A method for clustering objects for spatial data mining. *Knowledge and Data Engineering, IEEE Transactions on*, 2002. 14(5): p. 1003-1016.
- [5] Zhang, T., R. Ramakrishnan, and M. Livny. BIRCH: an efficient data clustering method for very large databases. in *ACM SIGMOD Record*. 1996. ACM.
- [6] Guha, S., R. Rastogi, and K. Shim. CURE: an efficient clustering algorithm for large databases. in *ACM SIGMOD Record*. 1998. ACM.
- [7] Guha, S., R. Rastogi, and K. Shim. ROCK: A robust clustering algorithm for categorical attributes. in *Data Engineering, 1999. Proceedings., 15th International Conference on*. 1999. IEEE.
- [8] Ester, M., et al. A density-based algorithm for discovering clusters in large spatial databases with noise. in *Kdd*. 1996.
- [9] Hinneburg, A. and D.A. Keim. An efficient approach to clustering in large multimedia databases with noise. in *KDD*. 1998.
- [10] Wang, W., J. Yang, and R. Muntz. STING: A statistical information grid approach to spatial data mining. in *VLDB*. 1997.
- [11] Wang, W., J. Yang, and R. Muntz. STING+: An approach to active spatial data mining. in *Data Engineering, 1999. Proceedings., 15th International Conference on*. 1999. IEEE.
- [12] Moore, A.W., Very fast EM-based mixture model clustering using multiresolution kd-trees. *Advances in Neural information processing systems*, 1999: p. 543-549.
- [13] Smith, A., et al., *Sequential Monte Carlo methods in practice*: Springer Science & Business Media. 2013.
- [14] Frey, B.J. and D. Dueck, Clustering by passing messages between data points. *science*, 2007. 315(5814): p. 972-976.
- [15] Zhang, X., et al. K-AP: generating specified K clusters by efficient affinity propagation. in *Data Mining (ICDM), 2010 IEEE 10th International Conference on*. 2010. IEEE.
- [16] Jain, A.K., *Data clustering: 50 years beyond K-means*. *Pattern recognition letters*, 2010. 31(8): p. 651-666.
- [17] Rodriguez, A. and A. Laio, Clustering by fast search and find of density peaks. *Science*, 2014. 344(6191): p. 1492-1496.
- [18] Jia, H., et al., A density-adaptive affinity propagation clustering algorithm based on spectral dimension reduction. *Neural Computing and Applications*, 2014. 25(7-8): p. 1557-1567.
- [19] Jin, W., et al., Ranking outliers using symmetric neighborhood relationship, in *Advances in Knowledge Discovery and Data Mining, Springer*. 2006. p. 577-593.
- [20] Ha, J., S. Seok, and J.-S. Lee, Robust outlier detection using the instability factor. *Knowledge-Based Systems*, 2014. 63: p. 15-23.
- [21] Cassisi, C., et al., Enhancing density-based clustering: Parameter reduction and outlier detection. *Information Systems*, 2013. 38(3): p. 317-330.
- [22] Zhu, Q., et al., A clustering algorithm based on natural nearest neighbor. *Journal of Computational Information Systems*, 2014. 10(13): p. 5473-5480.
- [23] Samaria, F.S. and A.C. Harter. Parameterisation of a stochastic model for human face identification. in *Applications of Computer Vision, 1994., Proceedings of the Second IEEE Workshop on*. 1994. IEEE.

Information Mining Projects Management Process

Sebastian Martins^{1,2}, Patricia Pesado^{1,3}, Ramón García-Martínez²

1. PhD Program on Computer Science, Computer Science School, National University of La Plata, Buenos Aires, Argentina

2. Information Systems Research Group, National University of Lanus, Buenos Aires, Argentina

3. III-LIDI. Computer Science School. National University of La Plata – CIC Buenos Aires, Argentina

martinssebastian@yahoo.com.ar, ppesado@lidi.info.unlp.edu.ar, rgarcia@unla.edu.ar

Abstract— Information Mining (also known as Knowledge Discovery Process) is a growing discipline in continuous expansion. Most of the progress accomplished, are focus on the development activities (i.e. those technical activities associated with the comprehension and adaptation of data, and the implementation of data mining algorithm). According to this conceptual framework, several process models were developed, which allow organizing and defining the set of tasks related to the development of information mining projects. These approaches omit the set of tasks oriented to the management and control of the process. In this paper, we propose a transversal management process to the development process currently in use in information mining projects. The proposed process focuses on removing existing gaps, providing an improvement on the project's maturity and quality levels.

Keywords-component; Information Mining Engineering; Project Management of Information Mining; Knowledge Discovery Process; Data Mining.

I. INTRODUCTION

The data mining term, is strongly bound to the concept of large database and goes back to the definition of searching algorithms of knowledge patterns [1]. However, today there are research lines in such fields as text mining [2], images mining [3], patterns in information stream mining [4], web mining [5], among others. In this context, our search line considers appropriate to use the term "Information Mining" [6] as a generic reference to any of the aforementioned types of mining. Based on that software engineering has been defined in SWEBOK [7] as "the application of a systematic, disciplined and quantifiable approach to the development, operation and maintenance of software, and the study of these approaches, i.e. the application of engineering to software"; it is agreed define Information Mining Engineering (IME) [8] as the application of a systematic, disciplined and quantifiable approach to the development of Information Mining Projects and the study of these approaches, i.e. the application of engineering to Information Mining. From this perspective, information mining engineering focuses on defining procedures to guide the development of an information mining project. Its main objective is to identify interesting patterns and relevant pieces of knowledge to the organization, ensuring their correct understanding, and providing reliable support for the decision-making process. To achieve this goal it is necessary to establish a set of activities that provide the overall project structure, supporting the development process. These guidelines should not only define the development aspects, but also it must identify those management activities associated with the project progress.

It is pointed out that the IME definition focuses on determining processes, and not in the specific technical characteristics of its implementation, such as defining the information mining processes [9] and the procedure to derive it from the business domain [10], identifying prospective families of data mining algorithms to implement and their combinations to obtain certain pieces of knowledge, rather than focusing on implementing a specific algorithm. Defining an IME is necessary to move from an artisanal development to a systematic, measurable, reproducible and disciplined development. The importance of these characteristics, are not only in the possibility of documenting the process, maintaining the traceability thereof, allowing to learn from it and reuse that knowledge to other projects, but also provides a set of tools that enables the mature development of the discipline, such as estimate time and cost, anticipating potential risks, needs and requirements that the project involves, among others.

In this context, the objective of this paper is to propose a new process model for IME projects. First, several approaches defining process model for IME are presented (section II). Section III presents the proposed solution, identifying the main structure of the management process. Section IV describes the existence gaps in the models previously mentioned in section II, identifying the solutions defined by the proposed model (Section IV). Finally, section V concludes the contributions of the proposed process model with respect to the previous approaches.

II. STATE OF THE ART

The first three sections briefly describe the structure of the most applied process models for information mining projects, which are focused on technical activities more related to data mining. These models are: KDD, SEMMA and CRISP-DM. Thereupon, a process model for software development projects oriented to the IME that is based on the best practices of software engineering projects is presented.

A. Knowledge Discovery in Databases (KDD)

The forerunner process model KDD [11], aims to provide a set of tools to systematize the data mining process and the artisanal process of hypothesis selection. The underlying objective to the concept of KDD is to differentiate data mining, understood as the activity of applying different algorithms on data in order to obtain patterns, than the process needed to generate knowledge from data. That is, understanding data mining as a subelement that integrates the general process of obtaining knowledge patterns. The steps involved in this process are: *Learning the application domain*: includes relevant

prior knowledge and the goals of the application; *Defining a target data set*: where discovery is to be performed; *Cleaning and Preprocessing*: includes basic operations such as removing noise or outliers, strategies for handling missing data fields, etc.; *Data reduction and projection*: involves finding useful features to represent the data depending on the goal of the task; *Choosing the function of data mining*: includes deciding the purpose of the model derived by the data mining algorithm; *Choosing the data mining algorithm(s)*: matches a particular data mining method with the overall criteria of the KDD process; *Data mining*: includes searching for patterns of interest through applying the data mining algorithms; *Interpretation*: includes interpreting the discovered patterns as well as possible visualization of the extracted patterns, removing redundant or irrelevant patterns, and translating the useful ones into terms understandable by users; and *Using discovered knowledge*: includes implementing or documenting the results.

B. SEMMA Methodology

The SEMMA [12] methodology was defined by the SAS Institute as a process used to reveal valuable information and complex relationships in large amounts of data. SEMMA consists of five stages: *Sample* the data by creating one or more data tables; *Explore* the data by searching for anticipated relationships, unanticipated trends, and anomalies in order to gain understanding and ideas; *Modify* the data by creating, selecting, and transforming the variables to focus on the model selection process; *Model* the data by allowing the software to search automatically for a combination of data that reliably predicts a desired outcome; and *Assess* the data by evaluating the usefulness and reliability of the findings from the data mining process.

C. CRoss Industry Standard Process for Data Mining (CRISP-DM)

CRISP-DM [13] is a widely used methodology focused on the developmental tasks involved during the whole process, considering slightly some activities related to the management of the project. The phases that make up the process model are: *Business Understanding*: aims to comprehend the objectives and requirements of the project from the business perspective, as well as identifying the data mining problem(s) and perform the project planning; *Data Understanding*: starts with an initial data collection and proceeds with a data analysis, identifying data quality problems and possible interesting subsets to apply different hypothesis; *Data Preparation*: covers all activities needed to generate the final data set from the raw; *Modeling*: covers the selection, configuration and implementation of various modeling techniques; *Evaluation*: the built model is analyzed to ensure that the business objectives are accomplished; and *Deployment*: covers those activities oriented to integrate or to document the knowledge gained.

D. Process Model for Software Projects Focused on the Development of Information Mining Systems

In [14] an integrated approach is proposed, essentially composed by CRISP-DM and the standard to develop conventional systems IEEE Std. 1074 (and it also has some elements from ISO 12207 and 15504). This proposal pointed

out which phases and activities from process models for software engineering project might be used for IME projects.

III. PROPOSAL OF PROCESS MODEL

After more than a decade using process models focused on development activities associated with data mining, and because of the high failure rate (greater than 60%) [15], being CRISP-DM the main methodology, several researches highlighted difficulties in the existent process model [16-18]. From the analysis performed over different existing proposals, arises the need to define a process model for IME projects, focusing on discovering relevant pieces of knowledge to support the decision-making process, which incorporates the management vision required to successfully achieve the development of the Project.

The proposed process model, called MoProPEI, consists of two sub-processes: development and management. The development process is a version based on CRISP-DM which reduce iteration necessity, modifying the order and structure of the activities, and includes a set of tasks (such as business and problem modeling, deriving the information mining process, etc.) in order to facilitate the comprehension and implementation of the project as well as quality level of the resultant products. The management process for IME projects provides a transversal layer to the development process, which offers several mechanism that improve the quality and maturity level of the resultant product(s) and process. Additionally, this sub-process incorporates different tools developed over the last years by the research group in order to cover the existence gaps. Since the beginning of the discipline, there has been an extensive effort to develop tools or techniques belonging to the development process. Taking this into account, this paper focused on describing the management process, identifying some techniques developed by the research group (section A), and explaining in detail its structure in section B.

A. Management Tools for IME

Because of the differences among IME and other projects, emerge the need to define a set of *ad hoc* tools / techniques that cover the existent conceptual gaps. During the last years, the research group has been working in a set of tools to resolve such necessity, among them:

Britos et Al. [19] describe a framework which defines tasks required for elicitation of requirements, and propose a reference model for identification and documentation of relevant concepts to carry out an IME project, along with the correlations between these concepts. This process consists of five steps which cover different aspects of the project, which are implemented throughout the process: in the phases of business understanding and modeling belonging to development process and in the initiation phase from management process.

Pytel et Al. [20;22] define cost and effort estimation method for information mining projects. This process describes the characteristics, and analyzes the possible values over which the effort and cost required to complete the project are obtained. These features are divided into three categories according to the origin of data to be analyzed: factors related to type of project, data and resources. This tool can be applied in

estimation and responsibilities activity, belonging to the planning phase from the management sub-process.

In [21;22] a method oriented to determine the project feasibility is described. This procedure identifies the set of characteristics to evaluate, which are classified in three categories: plausibility, adequacy and success. From these criteria it is determined whether the project is feasible, if IME is the best solution to the problem and whether the results can be useful for the organization. This tool can be applied in the assess situation activity (initiation phase).

Basso et Al. [23] define a set of metrics applicable to IME projects, grouped by three categories: data, models and project. In each category identifies tasks and characteristics to measure and their associated metrics. This tool can be applied in the estimation and responsibilities activity.

B. Management Process

The management process occurs as a supporting element to development process, in which the execution of its tasks is not linear, but is made based on project progress. This sub-process, structured by three additional levels (phases, activities and tasks), provides a higher level of granularity. It consists of five phases, each one composed by several activities (also referred as unity in this paper), identifying a set of tasks with a common goal. Figure 1 shows the phases that comprise it (at the left of the bracket in the picture) and for each phase the activities associated and their existing dependencies (in curved rectangles and lines respectively). In the top and bottom of the figure, there are activities from the development process, and inside each one is identified the phase and activity name which belongs (upper and lower section, respectively). Arrows represent interaction between two activities (dependent documents). There are also two types of lines ending in point: solid and dot, both represent an input document to an activity, which is produced by the team or the client respectively.

The first phase, named *Initiation*, contains four activities: "Communication Protocol" aims to determine formal channels of interaction, according to the actual possibilities among staff and experts in the business domain. Its inputs are customer speech (external), politics of contracting organization (external), politics of organization (internal) and project leader speech (internal), and produce as output the Communication Protocol Report (CPR); "Exploration of Initial Concepts" where the team makes a first approximation to the client characteristics and their needs. As result of the activity, initial exploration report (IER) is made, using as input the customer speech and the CPR; "Assess Situation" analyses the project characteristics, client requirements, existent risks and possible contingencies to determine the feasibility of the project. Its inputs are: existent resources (external), previous experiences of projects (internal) and IER. It generates five elements as results: guidelines projects report (GPR), internal and external resources reports, and project risks and feasibility report; and "Life Cycle Definition" determining the structure and iterations of the project, in order to encourage the development and success of the project. The requirements document (RD), generated in the activity business problem comprehension, and GPR are inputs over which the life cycle model is selected.

Planning is the second phase, composed by three activities: "activities planning", whose inputs are: the life cycle selected, the RD and GPR. Its aims are to select and to determine the time required to finish each task of the project, producing as outputs the activity calendar and map, and the list of metrics; "resources planning" whose inputs are activity calendar, business problems (identified in the activity business problem comprehension), RD, internal resources report and GPR. Its main purpose is to determine what resources are required and when (including outsourcing possibility), generating as outputs: the outsourcing plan, material resources plan, human resources plan and resources required report (RRR); "estimation and responsibilities" where the time and cost of the project, and the scope and obligations of the parties are determined from the RD, business objectives (generated in the business analysis activity belonging to the development process), report of external resources, project risks and feasibility reports, GPR, business problems, activity map, outsourcing plan and RRR.

Support composed by three activities: "Life Cycle Management" whose objectives are: to formalize the scope of each project iterations, and to establish the reached achievements, readjusting the scope of the next iteration if necessary. Their inputs are: the report of external resources, life cycle selected, material resources, outsourcing and HR plans and activity calendar and their outputs are: formalization of cycle start and formalization of cycle end; "Implementation Management" covers those activities related with the development of the product, such as defining the objectives to be performed in each step of the project, integrating internal products, communicating project progress, etc. Its inputs are: staff list (internal), outsourced product (external), project quality report, occurred risks report, activity progress report (these three last elements produced in the activities control unity), control of outsourced tasks (from Resources Control), business problem risks related (from Business Problem Understanding, development process), contingency plan, activity map, outsourcing plan and RRR, and produce as outputs: outsourced work Integration report (OWIR), corrective actions implementation report (CAIR), staff responsibilities report (SRP), human and material resources contracts, list of acquired resources (HR and materials), outsourced tasks report and outsourced tasks contract; and "Configuration Management" which aims to document the relevant information produced throughout the project, using as input: OWIR, CAIR, and formalization of cycle opening and ending.

The *Controlling and Quality* phase involves four activities oriented to improve the maturity of the project and its products, and keep tracking of all the elements required to complete successfully the project: "Resources Control" whose aim is verifying the accomplishment of the planning made, regarding the incorporation of resources (materials, products and human). Its inputs are: outsourced product, business problems and objectives, SRP, human and material resources contracts, list of acquired resources (HR and materials), outsourced tasks report, outsourced tasks contract, Outsourcing Plan and RRR; "Measurements" quantifies elements of interest for the project analysis, using as input the metrics list and the cost of the project activities (internal), generating the reports of metrics and costs.

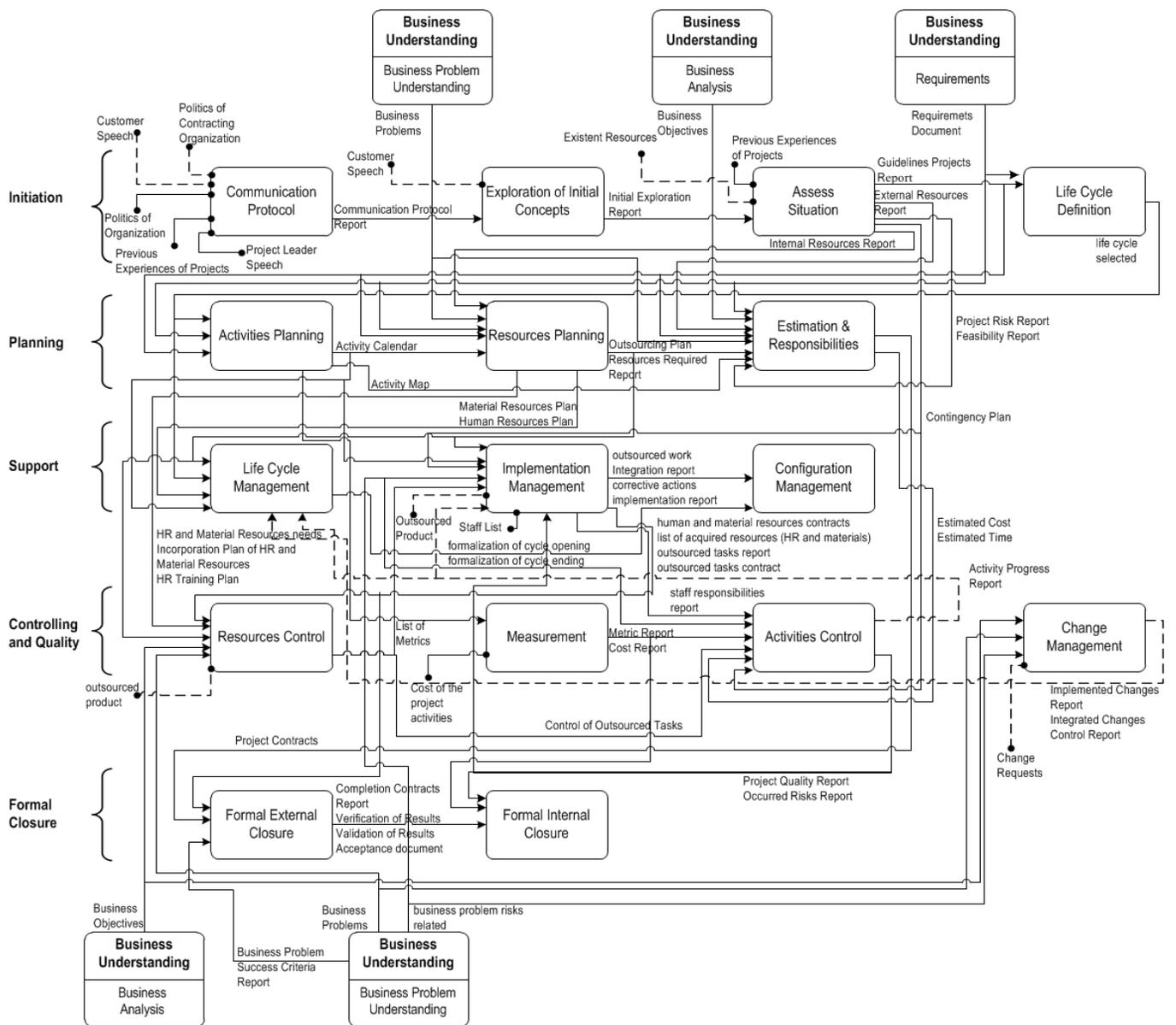


Figure 1. General Structure of the Management Process

“Activities Control” evaluates several project characteristics (progress of activities, time, cost, possible risks, quality, etc.) in order to identify possible deviations that jeopardize the successful development of the project. Its inputs are: SRP, business problem risks related, reports of metrics and cost, control of outsourced tasks, cost and time estimations and contingencies plan, producing as outputs: the reports of project quality, project risk and activity progress; and “Changes Management” which covers all activities aimed at evaluating, implementing and monitoring the changes required for the project, using the following inputs: change requests, business problems, business problem risks related and business objectives, and generates implemented changes and integrated changes control report.

The last phase *Formal Closure* is composed by two activities: “Formal External Closure” aims to ensure the achievement of the objectives required by the customer, ending

with the commitments between the parties involved in the project. Its inputs are: business problem success criteria report (from the business understanding problem activity which belongs to the development process), project contracts, human and material resources contracts, list of acquired resources, outsourced tasks report and outsourced tasks contract, and generates as outputs: completion contracts report, Verification and Validation of Results, and acceptance document; and “Formal Internal Closure” whose objective is to obtain useful knowledge from the characteristics analyzed throughout project, in order to be reused in future ones. Its inputs are: the outputs from the previous activity, metric and cost report, project quality report and occurred risks report.

IV. DISCUSSION

The proposed models, mainly focused on developmental activities, have more than fifteen years in the market and have

been applied in large quantities and varieties of projects. In recent years many issues and deficiencies have been identified. Neither KDD, nor SEMMA nor CRISP-DM define a process model focused on the management and control of the project, from this point of view our proposal attempt to cover this vacancy. Furthermore, both KDD, as SEMMA do not consider activities targeted to project management, only focusing on the developmental activities associated with business and data understanding, data selection and processing, implementation of data mining algorithms and analysis of results as pieces of knowledge. Although CRISP-DM identifies a set of activities related to project management, including: risks and contingencies, produce project plan, cost and benefit, Plan Deployment, Plan Monitoring and Maintenance and Review Project, it lacks of others extremely important features associated with the maturity level and success rate of projects.

Marban et Al. [14] define a scheme that identifies from the missing activities of their predecessors, which Software Engineering activities could be applied, however they do not clearly defined their objectives, scopes, inputs/outputs and possible techniques according to the objective of IME. Our proposal differs from this approach in that it is conceived considering as the general objective of the project: extracting reliable and relevant pieces of knowledge to support the decision-making process [8; 11; 24], and it attempts to deepen those aspects that are not defined in Marban’s proposal (describing in detail processes, phases, activities, tasks, inputs/outputs, techniques and their goals). Table I summarize the gaps in the previous processes, identifying whether the set of activities require for a IME project, grouped by the phases proposed in MoProPEI, are full, partially or not covered.

Table I. Comparison among processes models: management perspective

		PROCESS				
		KDD	SEMMA	CRISP-DM	MARBAN	MoProPEI
ACTIVITIES	Initiation	□	□	▒	▒	■
	Planning	□	□	□	▒	■
	Support	□	□	□	■	■
	Controlling and Quality	□	□	□	▒	■
	Formal Closure	□	□	▒	▒	■

Due to CRISP-DM is *de facto* standard in industry [14], holding for the past 10 years its presence as the main process model used by companies and data analysts [25] and the most robust approach, in the next paragraphs we describe the novelties with respect to CRISP-DM, identifying what activities from MoProPEI pretend to cover each of the aforementioned gaps:

Project Feasibility Analysis: determining success of the project at an early stage, avoiding taking risks which cannot be overcome or their cost is too high. The feasibility method introduced in section III.A, provides a guide that helps to make the decision of continue or not with the project according to its

characteristics. This method can be implemented in the activity *assess situation* in initiation phase.

Planning metrics: taking measurements and analyzing the results promotes the comprehension of project progress and product as well as its quality, using these results to improve the process to be performed and their resultant product(s). This deficiency can be resolved in the activity: activities planning by identifying and listing metrics applicable, implementing the tool for metric described in section III.A.

Planning and controlling project costs and time: monitoring these tasks allow to keep the fluctuation of these variables between preset values, being able to take early action if necessary. The activities: Estimation and Responsibilities, and Measurement, belonging to the phases “planning”, and “Controlling and Quality” respectively, consider these problems by the techniques: plan of cost and metrics and the report of metrics and cost.

Project documentation Management: due to the dynamic and iterative features of IME projects, controlling the generated documentation is essential in order to identify current version over what each member of the team is working on as well as maintain traceability of documents produced. Additionally, it enables the data analyst to reproduce any result, being able to return to a previous state of the project, while providing a higher level of maturity and quality in the project. Finally, controlling and generating new ideas from the experiences documented are simplified. The technique integrated changes control report (belonging to the change management activity) covers those problems.

Defining the responsibilities and obligations of the parties involved in the project: make a legal commitment between the parties involved in the project provides support for both sides, and also places on record and clarifies the scope and limitations of the project. This problem is resolved in the Estimation and Responsibilities activity, where the contracts required for the project are defined.

Analysis and selection of life cycle model: engineers identify the possible alternatives and choose the best option according to the characteristics of the project and information mining team. The project structure, order of execution and interaction between stages are defined, adjusting the time and cost to the project needs. The "life cycle definition" activity takes into consideration this problem.

Analysis and selection of the activities associated with the project: the engineers define activities and tasks necessary to successfully perform the project, according to the characteristics of the business domain, client requirements and information mining team. The "Planning activity", which belongs to the planning phase, helps to define the structure of the project (by activities map technique), providing better support for planning and development thereof, reducing its time and costs.

Identify and manage the outsourcing possibility: according to the characteristics of the project and the team, some parts or the whole project can be outsourced. Outsourcing involves performing a set of tasks and controls to ensure the project success. These tasks are considered throughout the sub-process

by the activities: assess the situation, where the outsourcing possibility is determined by generating the report possibility of outsourcing; Estimation and Responsibilities, where the scope and obligations of the parties under contracts are defined; and resources control and configuration management, where the team tracks the progress of outsourced item and performs the integration of the products obtained.

Planning and Control the resources distribution: anticipate human and material resources needs in different sections of the process, improves the development of the project. This problem can be solved applying the techniques: report of required resources and report of resource acquisition, implemented in the activities resource planning and resource control respectively.

V. CONCLUSIONS

The contributions provided by the proposed process against previous approaches are:

- [i]. We proposed a process model composed by two sub-processes (development and management). The management sub-process is conceived as a transversal layer to the development sub-process, on which the discipline has been focused on throughout its history, and its aim is to cover the current gaps in the areas of control and project management.
- [ii]. The process integrates a set of tools created *ad hoc* [9;10;19-23], oriented to reduce the risk and improve the maturity and quality level of the process and the resultant product(s).
- [iii]. The structure of management sub-process is described, identifying the set of activities required for IME project, detailing each activity objective and dependencies (inputs and outputs). The activities are grouped into phases, from the similarities in their goals for the overall process.

We identify as future research work:

- [i]. Extend the set of available techniques for different management activities, such as for the phase of requirements elicitation and analysis and evaluate how the proposed model can be used to reduce risks and improve productivity.
- [ii]. Developing experiments to compare effectiveness of the proposed process model against CRISP-DM.

REFERENCES

[1] Maimon, O. y Rokach, L. (Eds.). 2005. Data mining and knowledge discovery handbook. Springer.

[2] Tan, A. 1999. Text mining: The state of the art and the challenges. In Proc. PAKDD 1999 Workshop on Knowledge Discovery from Advanced Databases. pp. 65-70.

[3] Hsu, W., Lee, M., Zhang, J. 2002. Image mining: Trends and developments. Journal of Intelligent Information Systems, 19(1): 7-23.

[4] Gaber, M., Zaslavsky, A. Krishnaswamy, S. 2010. Data stream mining. En Maimon, O. and Rokach, L. eds. Data mining and knowledge discovery handbook. Springer, pp. 759-787.

[5] Kosala, R., Blockeel, H. 2000. Web mining research: A survey. ACM SIGKDD Explorations Newsletter, 2(1): 1-15.

[6] Gopal, R., Marsden, J., Vanthienen, J. 2011. Information mining: Reflections on Recent Advancements and the Road Ahead in Data, Text, and Media Mining. Decision Support Systems, 51(4): 727-731.

[7] Abran, A., Moore, J. W., Bourque, P., Dupuis, R., Tripp, L. 2004. Guide to the Software Engineering Body of Knowledge (2004 version). IEEE. ISBN 0-7695-2330-7.

[8] García-Martínez, R., Britos, P., Pesado, P., Bertone, R., Pollo, F., Rodríguez, D., Pytel, P., Vanrell, J. 2011. Towards an Information Mining Engineering. In Software Engineering, Methods, Modeling & Teaching. Medellín University Press. ISBN 9789588692326. pp. 83-99.

[9] García-Martínez, R., Britos, P., Rodríguez, D. 2013. Information Mining Processes Based on Intelligent Systems. Lecture Notes on Artificial Intelligence, 7906: 402-410. ISBN 978-3-642-38576-6.

[10] Martins, S., Rodríguez, D., García-Martínez, R. 2014. Deriving Processes of Information Mining Based on Semantic Nets and Frames. LNAI, 8482: 150-159. ISBN 978-3-319-07466-5.

[11] Fayyad, U., Piatetsky-Shapiro, G., Smyth, P. 1996. From data mining to knowledge discovery in databases. AI Magazine, 17(3): 37-54.

[12] SAS Institute Inc. 1998. SAS Institute White Paper, From Data to Business Advantage: Data Mining, The SEMMA Methodology and the SAS® System, Cary, NC: SAS Institute Inc.

[13] Chapman, P., Clinton, J., Keber, R., Khabaza, T., Reinartz, T., Shearer, C., Wirth, R. 2000. CRISP-DM 1.0 Step by step BI guide.

[14] Marbán, O., Mariscal, G., Menasalvas, E., Segovia, J. 2007. An Engineering Approach to Data Mining Projects. Lecture Notes in Computer Science, 4881: 578-588. Springer.

[15] Gondar, J.E. 2005. Data Mining Methodology. Data Mining Institute. ISBN: 978-84-96272-21-7.

[16] Wirth R., Hipp J. 2000. CRISP-DM: Towards a standard process model for data mining. Proceedings of the 4th International Conference on the Practical Applications of Knowledge Discovery and Data Mining, Manchester, UK, pp. 29-39.

[17] Yang Q., Wu X., 2006. 10 Challenging Problems in Data Mining research, International Journal of Information Technology and Decision Making 5(4), pp. 597-604.

[18] Lavrac N., Motoda H., Fawcett T., Holte R., Langley P., Adriaans P., 2004. Lessons Learned from Data Mining Applications and Collaborative Problem Solving. Machine Learning, 57. 13-34.

[19] Britos, P., Dieste, O. and García-Martínez, R., 2008, in IFIP International Federation for Information Processing, Volume 274; Advances in Information Systems Research, Education and Practice; David Avison, George M. Kasper, Barbara Pernici, Isabel Ramos, Dewald Roode; (Boston: Springer), pp. 139-150.

[20] Pytel, P., Britos, P., García-Martínez, R. 2013. A Proposal of Effort Estimation Method for Information Mining Projects Oriented to SMEs. Lecture Notes in Business Information Processing, 139: 58-74. ISBN 978-3-642-36610-9.

[21] Pytel, P., Britos, P., García-Martínez, R. 2013. Proposal and Validation of a Feasibility Model for Information Mining Projects. Proceedings 25th International Conference on Software Engineering and Knowledge Engineering. pp. 83-88. ISBN 978-1-891706-33-2.

[22] Pytel, P., Hossian, A., Britos, P., García-Martínez, R. 2015. Feasibility and Effort Estimation Models for Medium and Small Size Information Mining Projects. Information Systems Journal, 47: 01-14. Elsevier. ISSN 0306-4379.

[23] Basso, D., Rodríguez, D., García-Martínez, R. 2013. Proposal of Metrics for Information Mining Engineering Projects (In Spanish). Workshop on Data Bases and Data Mining. Proceedings XIX Argentine Congress of Computer Science. pp. 983-992. ISBN 978-987-23963-1-2.

[24] Witten, I. H., Frank, E., Hall, M. 2011. Data mining: Practical Machine Learning Tools and Techniques. 3rd ed. ISBN 978-0-12-374856-0.

[25] Kdnuggets. 2014. What main methodology are you using for your analytics, data mining, or data science projects? Poll (Oct 2014). <http://www.kdnuggets.com/polls/2014/analytics-data-mining-data-science-methodology.html> (last access 10/02/2016).

An investigation of students behavior in discussion forums using Educational Data Mining

Crystiano J. R. Machado, Bruno R. B. Lima and
Alexandre M. A. Maciel
University of Pernambuco - UPE
Recife, Brazil
{cjrm, brbl, amam}@ecomp.poli.br

Rodrigo L. Rodrigues
University Federal Rural of Pernambuco
Recife, Brazil
rodrigo.linsrodrigues@ufrpe.br

Abstract— Discussion forums are an important feature in the Distance Education courses, supporting learning and facilitating interaction between students and teachers. This work aims to investigate behavioral aspects of students in virtual learning environments using Educational Data Mining. It is proposed to use the K-Means clustering technique as a way to identify students with common patterns of behavior based on their interactions in the forums. This work achieves good results from the application of clustering technique for this particular issue.

Discussion Forums; Distance Education; Behavior Aspects; Educational Data Mining; Clustering.

I. INTRODUCTION

Demand for Distance Learning (DL) has shown growth rates increasing. The results of educational census conducted in Brazil between the years 2013 and 2014 show that the 309 institutions surveyed, there are a total of 4,044,315 students enrolled in distance learning courses. This number is expected to become even more significant in 2015, as 82% of these institutions enrollment will be even greater. These results guarantee the DL one leadership position among the types of education in Brazil [1].

Part of technological solutions used in distance education are contained in so-called Virtual Learning Environments (VLEs). According to Santana [2], the growth of distance education has motivated the academic community and boosted the development of scientific research. These studies are favored because of VLEs possess the ability to accumulate in your databases a lot of information that is valuable for the analysis of student behavior [3]. Majority of this information refers to the communication generated by the students through the existing tools in the DL environments such as discussion forums, chats and digital media postings.

According to Azevedo [5], the involvement of students in discussion forums is an important activity for building the knowledge. By analyzing the interaction of students in the forums, the teacher can diagnose relevant information about students. On the other hand, due to the large volume of data, the analysis of this information requires the help of specific tools.

In this context, a strategy widely used today is the Educational Data Mining (EDM). Bittencourt and Costa [4] define EDM as an emerging area of research that seeks to

develop, adapt and apply methods for the discovery of knowledge, in order to identify data sets from large information bases in educational systems.

The objective of this study is to present an investigation developed with students, teachers and tutors of undergraduate courses in distance education and intends to examine behavioral patterns of students concerning interactions in the forums of the analyzed disciplines as a way to identify groups with common characteristics. Therefore, this paper is organized as follows: section II presents the Theoretical Foundation; in section III is presented in detail the process of the Experiments; Section IV Results and Discussions are presented; in Section V are held the Conclusions about this work.

II. THEORETICAL FOUNDATION

A. Educational Data Mining

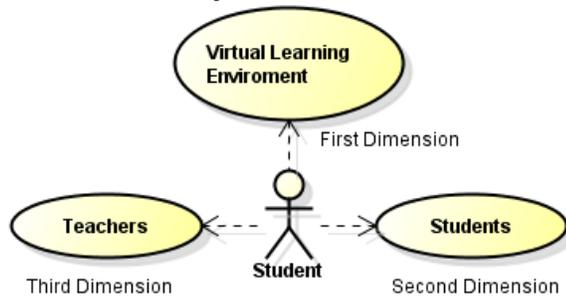
According to Webber, Zap and Lima [7], the Educational Data Mining (EDM) is an academic field of research that seeks to obtain, from the data stored in educational environments, new and useful information in order to develop and strengthen the cognitive theories of teaching and learning. More comprehensively, the "knowledge discovery can help teachers conduct their classes better, identifying difficulties, understanding better the learning process of students and improving teaching methods" [8].

For Manhães, Cruz, Costa, Zavaleta and Zimbrã [9], the EDM is related to the application of traditional techniques Data Mining in the several data fields of application in educational settings and that mostly arise from the Distance Learning environments. However, due to the existence of the information on different hierarchy levels, there are algorithms and tools used in Data Mining area can't be applied in the analysis of educational data without modification [10].

In parallel to the growth of research that is carried out in the area of EDM, so does the amount of data that is stored from the DL. For Romero, Ventura and García [8], this is due to inclusion of new features that are inserted in the Distance Learning platform. This enables interactions as: access to forums, questions and answers in areas that are designed to questionnaires and also due to direct communication between participants.

Gottardo, Kaestner and Noronha [6] propose that the representation of actions taken by a student in a Virtual Learning Environment (VLE) can be divided into three dimensions, seeking to contemplate the several aspects of use and interaction, as shown in Figure I.

FIGURE I. Representation of a Student Interaction in VLE.



- **First Dimension:** General profile of use of the VLE: this dimension aims to identify data representing characteristics of planning, organization and student time management for the course.
- **Second Dimension:** Student-Student interaction: the purpose of this dimension is to analyze characteristics that come to see if students interact with each other using the available tools of the VLE.
- **Third Dimension:** Student-Teacher interaction: this dimension aims to investigate how teachers or tutors interact with students in the context of VLE.

B. Cluster analysis

According to Baker [10], Educational Data Mining can be divided into five primary categories, one is called Clustering and aims to split the data into groups that share similar characteristics. This work focuses on the cluster analysis technique through *K*-Means algorithm.

According to Steinley [11], *K*-Means is one of the most used existing clustering methods in the literature, being defined as a partitioning method based on the definition of core elements of clusters, called centroids. Complementing this idea, [12] mentions that this partitioning occurs because the data set is divided into *K* subsets, where the value *K* is predetermined.

Among the main advantages of using the *K*-Means clustering technique as are its simplicity and speed. These characteristics favor its application in the process of mining of large databases. The disadvantage of the algorithm, one can consider the dependence of the parameter *K* represents the number of clusters on which data should be grouped together. In addition, the startup of the cluster centroids influences quality [12]. In order to minimize these negatives [13] suggests that the algorithm is executed with different values of *K* and boots.

According to Puma-Villanueva and Zuben [14], the main challenge in the realization of clusters is the analysis and definition of a great number of groups. Therefore, in this work, it adopted the technique Silhouette Index (SI) as a method of evaluation of clusters during the review process of the groups. The SI aims to evaluate and assign a grade to the quality of the result of the group, taking into account the distances inter and

intra groups. Well separated and compact groups have better quality.

In implementing the SI on the formed group, each group observation presents a value that varies in the interval [-1, 1]. The closer the value of a note is to 1, the smaller the distance (or dissimilarity) between the analyzed observation and any comments allocated to other groups. Thus, it can be considered that the individual was properly allocated in the current cluster. Moreover, SI values close to -1 indicates that the individual is allocated to an unsuitable group. Values near 0 indicate that the individual does not belong to a cluster or another [15].

C. Related works to this research

Currently, VLEs have assumed a prominent position in the academic and corporate education, and because of the resources offered by these environments, has been made possible synchronous and asynchronous communication between teachers, tutors and students participating in the course of Distance Learning [16]. As a result, several scientific works related to the EDM have been conducted.

Some of these works aimed to estimate the academic performance of students, as in [17], where an analysis is made using mining techniques student data in a VLE, reaching levels between 73% and 80% success in the performance estimates of the students. Moreover, [18] used a linear regression model in order to estimate the performance of students based on their virtual platform of learning interactions, taking into account lifestyle factors.

[19] Conducted experiments using decision trees algorithms in order to identify early graduate students the distance with risk avoidance, allowing the discovery of the disciplines that influence more in the avoidance of an undergraduate degree. In [20] presents an analysis of improvement of pedagogical actions, applying techniques of MDE in order to support behavior detection processes linked to truancy.

From the point of view of the interactions in distance education forums, [21] use statistical techniques and EDM in order to investigate the pattern of behavior of students and teachers of distance learning courses, using association rules and classification algorithms. [22] Use the semi-supervised algorithm *SVM-KNN* with the proposal to create new possibilities for the management and automatic monitoring of the forums on distance education courses. In turn, [23] use clustering techniques in forums publications to generate indicators for Educational Management.

III. EXPERIMENTS

In this work, Moodle data were collected, the Learning Management System (LMS) used more between higher education institutions offering distance education courses in Brazil [1]. These data were obtained considering some attributes related to forms of student interaction in the discussion forums. The completion of the experiments was based on data from 46 students of Educational Organization and Public Policy course, of the course Physical Education. The result points relevant factors that should be considered in order to assist teachers and tutors in the specific monitoring needs of each student.

May be considered that such interactions arise from the exchange of information among students who share the same VLE to carry out a course as well as the exchange of information between students and their teachers or tutors. Moreover, it is also regarded as a type of interaction the actions that a student performs in the system, involving only the student and the VLE.

The selected attributes for this work were based on the study realized by Gottardo, Kaestner and Noronha [17], which were associated with various attributes of Moodle to each of the three students interaction dimensions in a VLE [6]. Table I shows the attributes discussed in this work during the process of Data Mining.

TABLE I. LIST OF ATTRIBUTES (BASED ON [17]).

Dimension	Attribute
General Profile of Use of the VLE	1. Total number of posts held in forums.
	2. Number of posts of other participants read on forums.
	3. Total number of revisions in previous posts held in forums.
Student-Student Interaction	4. Number of answers posted in forums referring to postings from other students.
	5. Number of postings of other students doing student posting reference.
Bidirectional Interaction Student-Teacher	6. Number of student posts that had responses made by teachers or tutors of the course.
	7. Number of teacher posts or guardians who had responses made by the student.

A. Preprocessing

Starting the preprocessing of data, a two-dimensional table is built, containing the list of students associated with their respective values for each attribute. Then, the table was submitted to the preprocessing and data transformation, in order to improve the quality of the mining process. Such procedure involved the following steps:

- Analysis of raw data - In this stage, for each attribute, the statistical analysis of raw data collected from the database through SQL queries was held. This analysis was performed through the values processed by the Weka tool, so that they could be aware of the data set characteristics.
- Cleaning the data - this step was done filling the columns that did not have defined values for attributes. Instances in the table that did not have a value associated with a particular attribute, had zero value assigned by default.
- Identification of outliers - In this stage, for each variable, the identification of instances in the table of each class was held, seeking to inconsistent values when compared with the other set of data. For this, each of the entries in the tables were compared with the standard

deviation and the average value of the respective attribute.

- Removal of outliers - After the identification of inconsistent entries in the tables for each attribute considered in this research, adjustments were made in SQL queries.
- Standardization - this step was carried out adjusting the scale of values for each attribute by linear normalization. The values used in this study corresponded to the interval [0,1].
- Analysis of the transformed data - Finally, the new set of data representing the attributes individually in each class was again assessed by Weka. The use of the tool at this stage of the transformation process enabled the interpretation consistent with values.

B. Organization of scenarios

The scenarios examined in this study are based on the variation of the number of groups depending on the size of the interactions of the students. During the execution of the experiments, we observed that the implementation of K-Means for formations from eight groups, generated clusters with only one student. Therefore, we considered scenarios with the number of groups ranging from two (minimum) and seven (maximum). As this technique the "K" represents the number of groups, we have $K = \{2, 3, 4, 5, 6, 7\}$.

For each value of K, scenarios were considered containing the attributes of each of the three dimensions. This approach allowed for the analysis of each dimension individually and therefore identify groups with specific behavior patterns. Therefore, considering the variation of attributes and also the number of clusters, They analyzed a total of 24 scenarios. Such scenarios can be divided into four sets.

The first group of analyzed scenarios aims to investigate behavioral aspects of the students related to the VLEs usage profile. The second group of scenarios discussed in this paper aims to investigate behavioral aspects of the students related to interactions between students in the forums. The third group of scenarios analyzed aims to investigate behavioral aspects of the students related to the two-way interactions between students and teachers or tutors in the forums. The fourth group of simulations aims to investigate students' behavioral aspects in the discussion forums, based on three dimensions simultaneously.

C. Organization of scenarios

The results with the formation of the groups were subsequently subjected to analysis of Silhouette Index (SI). The SI was implemented to receive as input the distribution of the groups formed by Weka and the data with the attributes of each class. As a result, SI displays in percent the quality of clustering formed by the K-Means.

Was selected the scenario that had better quality percentage to determine the best number of groups (value of K), i.e., the best formation of groups by analyzing all dimensions simultaneously. The best scenario was chosen for the analysis

stage of groups and analysis of the behavioral aspects of students.

After identifying the best distribution of students according to the number of groups, it was possible to individually analyze each of the dimensions and, according to the attributes of each, draw a profile of groups based on their interactions.

IV. RESULTS AND DISCUSSIONS

The results obtained after the experiments are presented and discussed below. As a result of this research, analysis of the behavioral aspects of the students was based on the characteristics of the formed clusters.

For the 46 students of the discipline analyzed it was found that the best result in the formation of groups process, considering the set of attributes of the 1st, 2nd and 3rd dimension simultaneously, occurred with $K = 7$. In this scenario, the quality of clustering was estimated at 28,5%, as shown in Table II. These groups have respectively 1, 9, 3, 11, 2, 10 and 10 students.

TABLE II. CLUSTERING PROCESS QUALITY FOR EACH DIMENSION OF INTERACTION.

	Dimension			
	1st	2nd	3rd	1st, 2nd e 3rd
$K = 2$	36,34%	44,73%	41,95%	26,24%
$K = 3$	34,36%	45,91%	43,72%	21,63%
$K = 4$	27,44%	43,8%	41,4%	21,49%
$K = 5$	36,18%	39,63%	40,17%	25,93%
$K = 6$	30,49%	24,14%	37,95%	27,17%
$K = 7$	31,94%	18,76%	40,13%	28,5%

According to the statistical values related to the attributes for student's clusters were formed, it is possible to identify some important features for each group.

In Group 1, students are found with reasonable average of posts, with an average of 45,67, and satisfactory values regarding the amount of revisions posts, with an average of 15,17, representing students with second highest average in this attribute. Students in this group have a satisfactory profile of VLE usage and with good levels of interaction with each other. However, there is also the low amount of posts that are read by the students of this group, represented by average of 0,33 read messages. I notice also that in this group there is greater involvement by the students than teachers or tutors.

In Group 2 are the students with the highest average postings made in the forums and highest average revisions posts, with average respectively equal to 80,88 and 21,38. However, this group of students have values unsatisfactory as regards the number of read messages. Students of this group, as well as in Group 4, did not stand out in any of the characteristics related to the interactions student-student and Student-Teacher.

In Group 3, besides the unsatisfactory average of read messages, there is poor performance on the number of posts held and the lower middle revisions on posts, and the group with the worst results in both aspects analyzed. This group represents students with less interaction with distance learning platform and

unsatisfactory levels of interaction with other students, teachers and tutors.

In Group 4 students with satisfactory average posts are found (average equal to 39,83), but with low average interaction through read or revised posts. In this group, none of the interactions of the students stands out compared to the other groups. In addition, the interaction of the students of this group in teachers or tutors posts is the second worst result, compared with the other groups.

Group 5 is characterized by representing students with more interaction with distance education platform. Furthermore, the Group 5 is students with good interaction with other students, being the group with the highest average in both attributes, represented by average equal to 83,75 and 63,50 to the attributes related to the Student-Student Interaction. You can also see the good level of interaction of these students with teachers or tutors of the course.

The class of Groups 6 and 7 have a satisfactory average as regards the number of posts, with an average respectively equal to 68,89 and 70, and resemble the amount of unread posts, with values unsatisfactory for this feature. However, such groups differ in the average revisions in posts made by students, where students of the Group 6 have satisfactory values for this attribute (average equal to 11,67), unlike the students in the Group 7 (average equal to 4,33). The Group 6 has the distinction of the good level of interaction with teachers and tutors of the course, with an average of 20 posts that had responses made by teachers or tutors of the course, and 53 posts who had responses made by the student.

The similarity of the behavioral pattern was found of students forming groups 3, 4, 5 and 6 with the results presented by Silva *et al.* [24], where they investigated the actions of students in forums, chats and downloads using the Ward method. Our work is distinguished by applying the K-Means technique for training of students clusters in discussion forums and then validating the quality of the clusters through technical Silhouette Index. In addition, our study analyzes the interactions of students in three different dimensions, based on Gottardo, Kaestner and Noronha [17].

With the approach used in this study made it possible to obtain significant results in the identification of homogeneous groups of students (i.e., students of the same group have a relevant level of similarity on the aspects analyzed), as well as in-depth analysis of its features. The analysis of the formed clusters are useful for course coordinators and mentoring, as well as teachers and the own students, to the extent that point to important indicators for search of valuable experiences related to students' interactions in order to replicate them in other classes.

V. CONCLUSIONS

Discussion forums are an important feature in VLEs, for providing a support to the learning process that facilitates interaction by students, teachers and tutors in distance education environments. We can affirm that, through the forums, one can notice the individual participation of students, discussing the topics of the courses and contributing to the construction of collective knowledge.

Thus, this study aimed to conduct experiments using the K-Means clustering technique on data extracted from the Moodle forums on 46 students of Educational Organization and Public Policy course, of the course Physical Education in a higher education institution. The results were analyzed by the technique Silhouette Index, in order to identify the best training groups in their respective disciplines. Finally, the interpretation of the results to identify the behavioral aspects of the students was held, based on three dimensions of interaction: General Purpose Profile VLE, Student-Student Interaction and Student-Teacher Interaction Bidirectional.

Analyzing the students of Educational Organization and Public Educational Policy discipline, can form seven groups with very particular characteristics. Group 3 present unsatisfactory results in three dimensions, with students who have a very limited interaction profile. Some groups stand out in specific dimensions, the group 5 for the General Purpose Profile VLE interactions and the interactions student-students, and the group 6 over the Bidirectional Interactions Student-Teacher. The other groups of this course stand out in specific ways in each of the three dimensions.

The identification of groups of students from the interaction profile in the discussion forums are useful in order to provide teachers with an educational support that enables a deeper understanding of behavioral aspects of their students. The result of the identification of the characteristics of groups of students can help teachers plan their class better, identify the possible difficulties of students, better understand the learning process of students and improve teaching methods.

To obtain better results for future work, analyzes will be made that in addition to quantitative characteristics, incorporates qualitative aspects to obtain results that can infer possible learning problems of students, from the content of the interactions in the discussion forums. Another relevant aspect for future works concerns a proposal that incorporates assessment of external quality of the clustering, in order to identify an optimal clustering model.

REFERENCES

- [1] ABED. Associação Brasileira de Educação A Distância. Censo EaD.br: relatório analítico da aprendizagem a distância no Brasil 2013. 1 ed. Curitiba: Ibpex, 2014.
- [2] Santana, L. C. "Integração de um mecanismo de mineração de dados educacionais ao moodle". 2015. 85p. Dissertação (Mestrado em Engenharia de Computação) - Escola Politécnica de Pernambuco - POLI, Universidade de Pernambuco - UPE, Recife. 2015.
- [3] Mostow, J.; Beck, J.; Cuneo, A.; Gouvea, E.; Heiner, C. A Generic Tool to Browse Tutor-Student Interactions: Time Will Tell!. In AIED (pp. 884-886). Maio, 2005.
- [4] Bittencourt, I. I.; Costa, E. B. (2011) "Modelos e Ferramentas para a Construção de Sistemas Educacionais Adaptativos e Semânticos". Revista Brasileira de Informática na Educação, v. 19, p. 85-98.
- [5] Azevedo, B. F. T. "Análise das mensagens de fóruns de discussão através de um software para mineração de textos". Anais do XXII Simpósio Brasileiro de Informática na Educação, Aracaju. 2011b.
- [6] Gottardo, E.; Kaestner, C.; Noronha, R. V. "Avaliação de Desempenho de Estudantes em Cursos de Educação a Distância Utilizando Mineração de Dados". Anais do XXXII Congresso da Sociedade Brasileira de Computação. 2012.
- [7] Webber, C. G.; Zat, D.; Lima, M^a. de F. W. do P. Utilização de Algoritmos de Agrupamento na Mineração de Dados Educacionais. Revista RENOTE - Tecnologias na Educação. V 11. N^o. 1. Julho, 2013. ISSN 1679-1916.
- [8] Romero, C.; Ventura, S.; García, E. Data mining in course management systems: Moodle case study and tutorial. Computers and Education, 51 (1), pp. 368-384. 2008.
- [9] Manhães, L. M. B.; Cruz, S. M. S.; Costa, R. J. M.; Zavaleta, J.; Zimbrã, G. Previsão de Estudantes com Risco de Evasão Utilizando Técnicas de Mineração de Dados. Instituto de Ciências Exatas – Universidade Federal Rural do Rio de Janeiro (UFRRJ). 2011.
- [10] Baker, R. S. J. D. Data Mining for Education. McGaw, B., Peterson, P., Baker, E. (Eds.) International Encyclopedia of Education (3rd edition). Oxford, UK: Elsevier. 2010.
- [11] Steinley, D. Standardizing variables in K-meand clustering. In D. Banks, L. House, F. R. McMorris, P. Arabie, & W. Gaul (Eds.), Classification, clustering and data mining applications (p. 53-60). New York: Springer. 2004.
- [12] Malta, C. S. "Estudos de Séries Temporais de vento Utilizando Análises Estatísticas e Agrupamento de Dados". Tese de mestrado. Rio de Janeiro, RJ – Brasil, Fevereiro de 2009.
- [13] Souza, T. A. "Agrupamento de Séries temporais de Vento para Avaliação da Disponibilidade de geração de Usinas Eólicas". Universidade Federal do Rio de Janeiro. UFRJ, COPPE. Dissertação de Mestrado. 2008.
- [14] Puma-Villanueva, W. J.; Zuben, F. J. V. "Índices de validação de agrupamentos". Universidade Estadual de Campinas. Unicamp. Faculdade de Engenharia Elétrica e de Computação. FEEC. 2008.
- [15] Anzanello, M. J.; Fogliatto, F. S. Selecting the best clustering variables for grouping mass-customized products involving workers' learning. International Journal of Production Economics 130 (2), 268-276. 2011.
- [16] Maciel, A. M. A.; Rodrigues, R. L.; Carvalho, E. C. B. "Desenvolvimento de um Assistente Virtual Integrado ao Moodle para Suporte a Aprendizagem Online". In: III Congresso Brasileiro de Informática na Educação, 2014, Brasil. Anais do XXV Simpósio Brasileiro de Informática na Educação, 2014. p. 382-391. ISSN 2316-65330.
- [17] Gottardo, E.; Kaestner, C. A. A.; Noronha, R. V. "Estimativa de Desempenho Acadêmico de Estudantes: Análise da Aplicação de Técnicas de Mineração de Dados em Cursos a Distância". Revista Brasileira de Informática na Educação 22.01 p. 45. 2014.
- [18] Rodrigues, R. L.; Medeiros, F. P. A. de; Gomes, A. S. "Modelo de Regressão Linear aplicado à previsão de desempenho de estudantes em ambiente de aprendizagem". Anais do Simpósio Brasileiro de Informática na Educação. Vol. 24. No. 1. 2013.
- [19] Santos, R. N. dos; Siebra, C. A.; Oliveira, E. D. S. "Uma Abordagem Temporal para Identificação Precoce de Estudantes de Graduação a Distância com Risco de Evasão em um AVA utilizando Árvores de Decisão". Anais dos Workshops do Congresso Brasileiro de Informática na Educação. Vol. 3. No. 1. 2014.
- [20] Rigo, S. J.; Barbosa, J.; Cambruzzi, W. "Educação em Engenharia e Mineração de Dados Educacionais: oportunidades para o tratamento da evasão". EaD & Tecnologias Digitais na Educação. V. 2.3. p. 30-40. 2014.
- [21] Pequeno, H.; et al. "Uma Análise de Interação em Fóruns de EAD ". Anais do Simpósio Brasileiro de Informática na Educação. Vol. 25. No. 1. 2014.
- [22] Oliveira Júnior, Roberto L. de; Esmín, A. A. "Monitoramento Automático de Mensagens de Fóruns de Discussão Usando Técnica de Classificação de Texto". Anais do Simpósio Brasileiro de Informática na Educação. Vol. 23. No. 1. 2012.
- [23] Silva, L. A.; Trindade, D.; de Paula, C.; Pinto, S. et al. "Mineração de Dados em publicações de Fóruns de Discussões do Moodle como geração de Indicadores para aprimoramento da Gestão Educacional". Anais dos Workshops do Congresso Brasileiro de Informática na Educação. Vol. 4. No. 1. 2015.
- [24] Silva, R. E. D.; Ramos, J. L. C.; Rodrigues, R. L.; Gomes, A. S.; Fonsêca, J. A. V. "Mineração de dados educacionais na análise das interações dos alunos em um Ambiente Virtual de Aprendizagem". Anais do Simpósio Brasileiro de Informática na Educação. Vol 26. No. 1. 2015.

Empirical Evaluation of the ProcessPAIR Tool for Automated Performance Analysis

Mushtaq Raza
INESC TEC/Faculty of Engineering
of the University of Porto
Rua Dr. Roberto Frias, s/n 4200-465
Porto, Portugal
+351920055092
uomian49@yahoo.com

João Pascoal Faria
INESC TEC/ Faculty of Engineering of
the University of Porto
Rua Dr. Roberto Frias, s/n 4200-465
Porto, Portugal
+351225081400
jpf@fe.up.pt

Rafael Salazar
Tecnológico de Monterrey
Ave. Eugenio Garza Sada 2501 Sur Col.
Tecnológico C.P. 64849, Monterrey,
Nuevo León, Mexico
+528183582000
rafael.salazar@itesm.mx

Abstract— Software development processes can generate significant amounts of data that can be periodically analyzed to identify performance problems, determine their root causes and devise improvement actions. However, conducting that analysis manually is challenging because of the potentially large amount of data to analyze and the effort and expertise required. ProcessPAIR is a novel tool designed to help developers analyze their performance data with less effort, by automatically identifying and ranking performance problems and potential root causes. The analysis is based on performance models derived from the performance data of a large community of developers. In this paper, we present the results of an experiment conducted in the context of Personal Software Process (PSP) training, to show that ProcessPAIR is able to accurately identify and rank performance problems and potential root causes of individual developers so that subsequent manual analysis for the identification of deeper causes and improvement actions can be properly focused.

Keywords- *Automatic Performance Analysis; Performance Analysis Tool; Personal Software Process; Empiric Assessment.*

I. INTRODUCTION

Development processes making intensive use of metrics and quantitative methods, such as the Team Software Process (TSP) [1] and Personal Software Process (PSP) [2], can generate large amounts of data that can be periodically analyzed by developers to identify their performance problems, determine root causes and devise improvement actions [3]. Although tools exist to automate data collection and produce performance charts and reports for manual analysis of TSP/PSP data [4][5][6], practically no tool support exists to automate developer performance analysis. The manual analysis of performance data for determining root causes of performance problems and devising improvement actions is challenging because of the amount of data to analyze [3] and the effort and expertise required.

To address those shortcomings, in previous work [7][8] we developed models, techniques, and tools to automate the

analysis of performance data produced in the context of high maturity software development processes. The developed ProcessPAIR tool, available freely in <http://blogs.fe.up.pt/processpair/>, is able to automatically identify and rank performance problems and potential root causes of individual developers in their performance data.

In the current paper, we focus on the empirical assessment of ProcessPAIR approach and tool, through a case study in which we analyze the performance data and analysis reports produced by several PSP trainees at Tecnológico de Monterrey, Mexico. Specific objectives and research questions are presented in Section III.

Section II presents some background information on our approach and tool. Sections III, IV, V and VI present the case study planning, performance model preparation, performance analysis, and results and discussion. Section VII presents some related work and section VIII presents final conclusions and future work.

II. BACKGROUND

Our approach involves three main steps (see Figure 1):

1. *Define*: Process experts define the structure of a performance model (PM) suited for the development process under consideration (TSP, PSP, or other). In our approach, a PM comprises a set of performance indicators (PIs) organized hierarchically by cause-effect relationships [8]. Examples are given in section IV.

2. *Calibrate*: The PM is automatically calibrated based on the performance data of many process users. The statistical distribution of each PI and statistical relations between PIs are computed from the data set [8].

3. *Analyze*: Once a PM is defined and calibrated, the performance data of individual developers can be automatically analyzed with ProcessPAIR, to identify and rank performance problems and root causes.

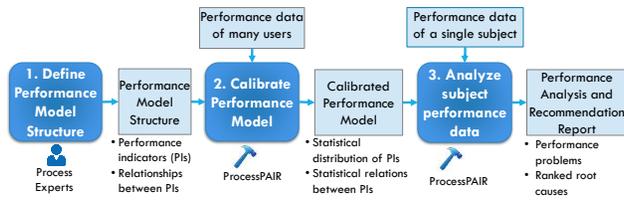


Figure 1. The ProcessPAIR approach.

Further details about each step are given next.

A. Model definition

The first step in our approach is the definition (as a tool extension) of the following elements of the PM:

- list of relevant PIs, including a formula for its computation from base measures;
- subset of top-level PIs;
- cause-effect relationships between PIs, determined by a formula or statistical evidence;
- sensitivity coefficients [9] between PIs related by a formula.

The performance model defined for analyzing PSP performance data will be presented in section IV.A.

B. Model calibration

The PM is automatically calibrated by ProcessPAIR from training data sets, generating the following data:

- approximate statistical distribution of each PI, represented by a cumulative distribution function;
- recommended performance ranges for each PI;
- sensitivity coefficients between PIs not related by an exact formula.

The approximate cumulative distribution function of each PI is computed by linear interpolation between a few percentiles computed from the training data.

Performance ranges are needed for classifying values of each PI of a subject under analysis into three categories: green - no performance problem; yellow - a possible performance problem; red - a clear performance problem. Such ranges are calibrated automatically from the training data, so that there is an approximately even distribution of data points by the colors.

Sensitivity coefficients between PIs not related by an exact formula are computed by first determining a linear regression equation from the training data and subsequently computing the corresponding sensitivity coefficient.

The data set used for calibration in the case study will be described in section IV.B.

C. Performance analysis

Having defined and calibrated the performance model, the performance data of individual developers can be automatically analyzed by ProcessPAIR, to identify and

rank performance problems and potential causes of individual developers. The results of the analysis are presented in multiple views, as shown in section V.B.

III. CASE STUDY PLANNING

D. Objectives and research questions

The overall objective of the case study is to assess whether ProcessPAIR is able to accurately identify performance problems of individual developers and their potential causes in the context of PSP training so that subsequent manual analysis for the identification of deeper causes and remedial actions can be properly focused and effort can be saved.

In PSP training, students develop a sequence of projects, with the stepwise introduction of the following practices: performance measurement (based on size, effort and defects); size and effort estimation; coding standards; design and code reviews; design templates and design verification; quality management [2].

More specifically, the goal of the case study is to answer the following research questions:

- **RQ1 (problem identification):** Is it possible to automatically analyze the performance data of an individual PSP developer in order to identify performance problems, with similar results but less effort than in manual analysis?
- **RQ2 (root cause identification):** Is it possible to automatically analyze the performance data of an individual PSP developer in order to determine the root causes of the identified performance problems, with similar results but less effort than in manual analysis?

E. Performance data under analysis

The subject data under analysis is based on a data set from Tecnológico de Monterrey, in Mexico, referring to 10 subjects (students) that developed 6 projects each using the PSP, in the scope of the “Software Quality and Testing” course in 2015. The subjects used Process Dashboard (<http://www.processdash.com/>) for collecting the standard PSP base measures. In the end of the sequence of projects, the subjects analyzed their personal performance in those projects and documented their findings and improvement proposals in a Final Report (written in Spanish).

F. Performance analysis procedures

Two of the authors of this paper, not involved in PSP training in Tec de Monterrey, both fluent in English and one with a good reading understanding of Spanish, translated into English and analyzed the final reports (in both English and Spanish), in order to extract relevant information for comparison with the tool-based analysis. Results from the tool-based analysis for each subject were effortlessly obtained by uploading the performance data

stored in Process Dashboard to ProcessPAIR. The extracted results from the final reports and from the tool for each subject were then collected into an appropriate table, as illustrated in Section VI. Subsequently, the results were classified according to the categories defined in Section V and statistics were computed shown in VI.

IV. PERFORMANCE MODEL PREPARATION

A. Performance model definition

To best fit the specific context of PSP training in Tec de Monterrey, we used the PSP performance model defined in our previous work [8] with minor changes. The full set of top-level and nested PIs can be seen in the first column of Figure 2. We consider three top-level PIs regarding predictability, quality, and productivity.

The major predictability PI in the PSP is the *Time Estimation Accuracy*, which we measure by the ratio between actuals and estimates. Since in the PSP's PROBE estimation method [2], a time (effort) estimate is obtained based on a size estimate of the deliverable (in added or modified size units) and a productivity estimate (in size per time units), we consider that the *Time Estimation Accuracy* is affected by the *Size Estimation Accuracy* and the *Productivity Estimation Accuracy*. Hence, the latter PIs are presented in Figure 2 as child nodes of the *Time Estimation Accuracy*. The rationale for further drilling down the *Productivity Estimation Accuracy* can be consulted in [8].

Product quality is usually measured by post-delivery defect density [10]. However, since the scope of the PSP is the development of small programs or components of large programs, post-delivery defects are seldom known. The PSP proposes an aggregated quality measure—the *Process Quality Index (PQI)*—that constitutes an effective predictor of post-delivery defect density [2][11]. Hence, we use the PQI as the top-level quality indicator to analyze. The PQI is computed based on five components, which are presented in Figure 2 as factors that affect the PQI. The exact formula can be consulted in [8], as well as the rationale for further drilling down these PIs.

In the PSP, productivity is usually measured in lines of code per hour, in spite of known limitations [10]. Since in the PSP time is recorded per process phase, when a productivity problem is encountered one can analyze the productivity per phase, to determine the problematic phase(s). Hence, Figure 2 shows a set of PIs for the productivity per phase, which together affect the overall productivity. Exact formulas can be consulted in [8], as well as the rationale for further drilling down these PIs.

B. Performance model calibration

To calibrate the performance model, we used a large PSP data set from the Software Engineering Institute (SEI) referring to 31,140 projects concluded by 3,114 engineers

during 295 classes of the classic PSP for Engineers I/II training courses running between 1994 and 2005. In this training course, targeting professional developers, each engineer develops 10 small projects. The calibration is performed automatically by the tool; the user has just to provide an input file with the data set.

V. PERFORMANCE ANALYSIS

A. Manual performance analysis

Regarding RQ1 (problem identification), based on the information available in the final report of each subject, we produced a table with a synthesis of cases in which the subject explicitly indicated bad performance or good performance in a PI for a specific project or overall (summary). Examples are shown in Table III, together with the support citations extracted from the final report.

Regarding RQ2 (root cause identification), for each case in which the subject explicitly indicated bad performance and corresponding causes, we filled in an additional column with the causes mentioned by the subject, as illustrated in Table III.

B. Automatic performance analysis

The results from the tool-based performance analysis for each subject were effortlessly obtained by uploading the performance data stored in Process Dashboard to ProcessPAIR. The results of the analysis are presented by the tool in multiple views.

The relevant view for problem identification is the *Table View*. This view presents the detailed evaluation of all PIs for all projects of the subject under analysis, as depicted in Figure 2. Each cell is colored green, yellow or red, in case, its value suggests no performance problem, a potential performance problem, or a clear performance problem, respectively. A cell is colored green (red) if its value lies within the range of the best (worst) 1/3 values in the calibration data. Cells with missing data are left blank. For example, the red cells in Figure 2 suggest that the main problems with time estimation accuracy occur in projects P2, P5, and P6. By expanding the nodes in this view, one can drill down to lower level PIs, following the hierarchical structure of the performance model, in order to identify potential causes of performance problems. For example, the red colored cells in Figure 2 suggest that the time estimation problem in P2 is caused by a size estimation problem.

The *Diagram View* (see Fig. 3) helps identifying and prioritizing, project by project, the causes of performance problems. The child indicators are sorted according to the value of a ranking coefficient representing a cost-benefit estimate that relates the cost of improving the value of the child indicator with the benefit on the value of the parent indicator [8]. For example, the diagram of Fig. 3 suggests

that the major cause for the poor productivity in project 5 is the poor productivity in the Design phase, followed by the Design Review, Plan, and Code phases.

Figure 2. An example of problem identification (Table View).

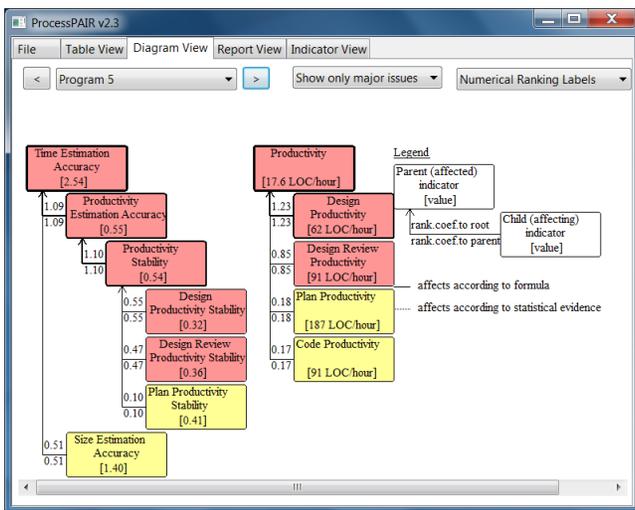


Figure 3. An example of root cause identification (Diagram View).

C. Comparison

The results extracted from the performance analysis report (manual analysis) and from the tool (automatic analysis), were compared as illustrated in Table III.

Regarding problem identification (RQ1), we considered that there is a match when the developer explicitly indicated

bad performance and the tool indicated a clear (red) or potential (yellow) performance problem. False positives occur when the developer explicitly indicates good performance, but the tool indicates a clear or potential performance problem. False negatives occur when the developer explicitly indicates bad performance, but the tool indicates no performance problem.

Regarding root causes identification (RQ2), we encountered three kinds of situations:

- *same causes*: the developer and tool indicate the same causes;
- *deeper manual analysis*: the tool accurately points out intermediate causes, and the developer points out deeper causes;
- *faults in manual analysis*: the developer overlooked important causes or pointed out erroneous causes.

VI. RESULTS AND DISCUSSION

A. Statistics

From the table produced in the previous step (as illustrated by the excerpts in Table I), we computed the statistics shown in Table I and Table II.

Regarding problem identification (RQ1), among the 187 cases analyzed, there are 180 matches (96%), with only 6 false positives (3%) and 1 false negative (1%). The false positives and the false negative correspond to boundary situations similar to the one illustrated in Table I.

Regarding root causes identification (RQ2), Table II is self-explanatory. From the 116 cases in which the developers explicitly indicated bad performance, only in 52 cases (with some examples in Table III) the developers pointed out root causes.

TABLE I. PROBLEM IDENTIFICATION STATISTICS.

		Automatic analysis		
		Green	Yellow	Red
Manual Analysis	Bad	1 (1% false negative)	45 (match)	70 (match)
	Good	65 (match)	6 (3% false positives)	0

TABLE II. ROOT CAUSES IDENTIFICATION STATISTICS.

Classification	Absolute frequency	Relative frequency
Same causes (<i>tool benefit: eliminate manual effort</i>)	10	19%
Developer pointed out deeper causes (tool pointed out intermediate causes) (<i>tool benefit: reduce manual effort</i>)	28	54%
Faults in manual analysis (<i>tool benefit: prevent user errors</i>)	14	27%
Total	52	100%

TABLE III. EXCERPTS OF THE PROBLEM AND ROOT CAUSES IDENTIFICATION AND COMPARISON TABLE.

Data point		Manual performance analysis		Automatic performance analysis		Comparison	
Top-level Indicator	Subject, Project	Problem identification	Root causes identification	Problem identification	Root causes identification (with ranking coefficient)	Problem identification	Root causes identification
Time Estimation Accuracy	S1, P2	Bad performance: “there are two that stand out for being very large, the program 2 and (...)”	“I attribute the bad (time) estimation to the (bad) size estimation”	Red	Size Estimation Accuracy (1.0)	Match	Same causes
Time Estimation Accuracy	S1, P1	Good performance: “the closest estimates were the program 1 and 4, with 18.90% and 17.20%”	-	Green	-	Match	-
Defects Injected	S1, Summary	Bad performance: “almost all programs have 50 to 100 errors per KLOC, which can improve”	“most defects injected in Design (mainly of type Function), followed by Code”	Yellow	Defects Injected in Design (12.4), Defects Injected in Code (0.8)	Match	Deeper manual analysis ⁽¹⁾
Size Estimation Accuracy	S2, P6	Bad performance: “the last program fall out of the (desired) range of 10% error”	-	Green	-	False negative ⁽²⁾	-
Time Estimation Accuracy	S3, P5	Good performance: “for program 4 and up (...) the estimation error is approximately within a range of -10% to 10%, which (...) is a good range”	-	Yellow	-	False positive ⁽³⁾	-
Productivity	S4, P4	Bad performance: “Program 4 represents a significant (productivity) downward”	“due to some defects (...) not identified in time, resulting in a time consuming testing phase”	Red	Code Productivity (62.7), Code Review Productivity (4.8)	Match	Faults in manual analysis ⁽⁴⁾

(1) The manual and automatic analysis coincide regarding the identification and prioritization of the problematic defect injection phases, with the quantitative prioritization in the automatic analysis. Additionally, the developer point out the most problematic defect type (Function).

(2) This false negative corresponds to a boundary situation. The actual size estimation error was approximately -15%, which is the threshold considered by the tool to distinguish green and yellow regarding size estimation. On the other hand, the developer considered an abnormally tight range of +-10%.

(3) This false positive corresponds to a boundary situation. The actual time estimation error was approximately -13%, which is the threshold considered by the tool to distinguish green and yellow regarding time estimation. On the other hand, the developer was ‘benevolent’ in his analysis, by considering -13% to be approximately within the +-10% range.

(4) Data shows that defects were removed in Code Review, not in Unit Test, and that much more time was spent in Code Review than in Unit Test.

B. Answers to the research questions

Regarding RQ1 – “Is it possible to automatically analyze the performance data of an individual PSP developer in order to identify performance problems, with similar results but less effort than in manual analysis”, we conclude that, in this case study, the automatic analysis produces similar results (without essentially any manual effort), with very few false positives (3%) and false positives (1%) corresponding to boundary situations.

Regarding RQ2 – “Is it possible to automatically analyze the performance data of an individual PSP developer in order to determine the causes of the identified performance problems, with similar results but less effort than in manual analysis”, the results in Table II show that, in the cases in which the manual analysis was not faulty (we found faults in 27% of the cases!), the tool-based analysis was able to point out the same causes as the ones found by the developers in their manual analysis (19% of the cases) or was able to point out intermediate causes in the same direction as the deeper causes identified in manual analysis (54%) of the cases. Hence, regarding RQ2, we conclude that the automatic analysis was able to identify either the same causes or causes in the same direction as the manual analysis.

Overall, the benefits of the tool-based analysis are:

- it can correctly identify the performance problems, saving manual effort;
- it can correctly identify causes for the identified performance problems, so that subsequent manual analysis for searching deeper causes can be properly focused, reducing the overall manual effort needed and the errors in manual analysis.

C. Limitations and threats to validity

In the case study presented, the conclusions obtained by the model-based analysis are very close to the ones obtained by the developers in their manual analysis. This suggests that our approach can be helpful in performance analysis and process improvement, by pointing out the areas to focus on manual analysis. However, further experiments need to be conducted to quantify the effort savings that can be achieved by conducting performance analysis with the help of our tool from the beginning.

Although our approach and tool are general and can be instantiated for any development process, the model and experiment described in this paper refer only to PSP performance data. We intend to replicate our approach to other development processes without having such a well-

defined measurement framework as the PSP, but we expect to encounter difficulties regarding data availability, data quality, and standardization.

VII. RELATED WORK

Our approach draws inspiration from existing work on process performance models (PPM) [9][12], benchmark-based approaches for software product evaluation [13], and defect causal analysis (DCA) techniques [14].

In the context of the CMMI process improvement framework, a PPM is a description of the relationship among attributes of a process or sub-process and its outcomes, developed from historical performance data, and calibrated using collected process and product measures [15]. The main difference is that our performance model conveys additional elements needed to identify performance problems (in the outcomes) and rank potential root causes (factors): recommended ranges for each PI; approximate statistical distribution of each PI; sensitivity coefficients (derived from exact or regression equations).

In our approach, in order to enable the automated identification of performance problems, after deciding on the relevant PIs, one has to decide on the relevant ranges. Our approach for defining such ranges draws inspiration from the benchmark-based approach developed by researchers of the Software Improvement Group [13][16] to rate the maintainability of software products, with adaptations for process evaluation instead of product evaluation.

The DCA approach [14] is essentially complementary to our approach. The main advantage of our approach is that it has the potential to identify relevant performance problems and causes in a fully automatic way, so that subsequent manual activities can be conducted in a more focused and efficient way, to further determine root causes and devise improvement actions.

VIII. CONCLUSIONS AND FUTURE WORK

The results of the case study show that the ProcessPAIR tool is able to accurately identify performance problems of individual PSP developers and potential causes for those problems. Hence, subsequent manual analysis for the identification of deeper causes and remedial actions can be properly focused, reducing the overall effort and possible errors in performance analysis.

As future work, we plan to build a comprehensive catalogue of improvement actions to recommend for the highest-ranked causes, build similar models for analyzing performance data produced in the context of other development processes, and conduct further experiments for assessing the effort gains with our tool.

ACKNOWLEDGMENTS

The authors would like to acknowledge the SEI and Tec de Monterrey for facilitating the access to the PSP data for performing this study. This work is partially financed by the ERDF – European Regional Development Fund through the Operational Programme for Competitiveness and Internationalisation - COMPETE 2020 Programme within project «POCI-01-0145-FEDER-006961», and by National Funds through the FCT – Fundação para a Ciência e a Tecnologia as part of project UID/EEA/50014/2013 and research grant SFRH/BD/85174/2012.

REFERENCES

- [1] Davis, N., and Mullaney, J. 2003. *The Team Software Process (TSP) in Practice: A Summary of Recent Results*. CMU/SEI-2003-TR-014.
- [2] Humphrey, W. 2005. *PSPsm: A Self-Improvement Process for Software Engineers*. Addison-Wesley Professional.
- [3] Burton, D. and Humphrey, W. 2006. Mining PSP Data. In *TSP Symposium 2006 Proceedings*.
- [4] The Software Process Dashboard Initiative home page. <http://www.processdash.com/>.
- [5] Philip, J., Kou, H., Agustin, J., Christopher, C., Moore, C., Miglani, J., Zhen, S., Doane, W. 2003. Beyond the Personal Software Process: Metrics Collection and Analysis for the Differently Disciplined. In *ICSE 2003*. Portland, Oregon.
- [6] Shin, H., Choi, H., and Baik, J. 2007. Jasmine: A PSP Supporting Tool. In *Proc. of the Int. Conf. on Software Process (ICSP 2007)*, LNCS 4470, Springer-Verlag, 73-83.
- [7] Raza, M., Faria, J. 2014. A Model for Analyzing Estimation, Productivity and Quality Performance in the Personal Software Process. In *Proc. of the 2014 Int. Conf. on Software and System Process (ICSSP 2014)*, ACM, 10-19.
- [8] Raza, M., Faria, J. 2015. A Model for Analyzing Performance Problems and Root Causes in the Personal Software Process. *Journal of Software: Evolution and Process*, John Wiley & Sons
- [9] Saltelli, A., Chan, K., Scott, E. M. 2008. *Sensitivity Analysis*, Wiley.
- [10] Jones, C. 2010. *Software Engineering Best Practices: Lessons from Successful Projects in the Top Companies*. McGraw-Hill.
- [11] Humphrey, W. 2009. *The Software Quality Profile*. White Paper, SEI.
- [12] Tamura, S. 2009. Integrating CMMI and TSP/PSP: Using TSP Data to Create Process Performance Models. CMU/SEI-2009-TN-033.
- [13] Alves, T., Ypma, C., Visser, J. 2010. Deriving Metric Thresholds from Benchmark Data. In *2010 IEEE International Conference on Software Maintenance (ICSM)*, 1-10.
- [14] Card, D.N. 2005. Defect Analysis: Basic Techniques for Management and Learning. *Advances in Computers*, vol. 64, 259-295, Elsevier.
- [15] Chrissis, M. B., Konrad, M., Shrum, S., 2003. *CMMI: Guidelines for Process Integration and Product Improvement*, 2nd Edition. Addison-Wesley.
- [16] Alves, T. 2012. *Benchmark-based Software Product Quality Evaluation*. Ph Thesis. U. Minho

Embedded Emotion-based Classification of Stack Overflow Questions Towards the Question Quality Prediction

Amit Kumar Mondal Mohammad Masudur Rahman Chanchal K. Roy
University of Saskatchewan, Canada
{amit.mondal, masud.rahman, chanchal.roy}@usask.ca

Abstract—Software developers often ask questions in Stack Overflow Q & A site, and their posted questions sometimes do not meet the standard guidelines. As a consequence, some of the questions are edited by expert users, some of them are down-voted, or some are even deleted permanently. Besides, the users (i.e., developers) might not get the expected solutions for their problems. In this paper, we study up-voted and down-voted questions from Stack Overflow, and analyze the relationship of embedded emotions with question quality. We use Sentiment140 API for identifying embedded emotions in the question texts, and then apply Feed-Forward Multilayer Perceptron (MLP) and Support Vector Machine (SVM) on the emotion data for developing a quality prediction model. Experiments using 38,920 Stack Overflow questions suggest about 70% precision and about 74% recall for our model with 10-fold cross-validation, and these findings clearly reveal the impact of human emotions upon the quality of a question.

Keywords – *Sentiment analysis; question quality; machine learning; StackOverflow*

I. INTRODUCTION

Software developers often search for solutions in the web for their encountered problems. Programming Q & A sites host millions of programming related questions that are answered by expert individuals from the community. Stack Overflow (hereby SO) is one of the most widely used programming Q & A sites that helps the developers harvest knowledge on programming problems and solutions. It contains millions of questions and answers, and has a median response time of 11 minutes [18]. However, due to poor quality in the question content, the users (i.e., developers) might not get the appropriate answers, and the questions might need to be edited by the experts, or it could be down-voted or even deleted by moderators. Asaduzzaman et al. [3] suggest that unanswered questions are increasing rapidly. They report about 299K questions that were posted during 2008 to 2012 and still are unanswered. Rahman and Roy [18] report that up to February, 2015, about 27% of StackOverflow questions were unresolved. Correa and Sureka [6] identify about 293K questions that were posted during 2009 to 2013 and were later deleted due to off-topic or low quality content. Thus, question quality is a major concern for Stack Overflow. The site also attempts to ensure the content quality standard and minimize the noise by setting up a clear guideline¹

¹<http://stackoverflow.com/help/how-to-ask>

for asking questions. However, the volume of the question is rapidly increasing, and manual checking and assurance of the questions' quality is prohibitively costly and highly challenging. One way to handle this challenge is to employ an automatic technique that can predict the quality of a question reliably during its submission. Based on our preliminary observation on 1000 question samples, we believe that human emotions or sentiments embedded in the texts of a question might affect the perception about the quality of the question. Too much negative words (i.e., pessimistic views) in the text of a question might increase the possibility of getting the question down-voted or deleted permanently or at least the question needs to be edited by the experts. This leads us to consider a special aspect, the embedded emotions or sentiments of an about-to-post question's text for predicting its quality. We thus attempt to answer the following two research questions in this paper:

- **RQ₁**: Is quality of a question affected by its author's emotions/sentiments embedded in the texts?
- **RQ₂**: Can a predictive model effectively predict the quality of the question based on the sentiment metrics?

Existing studies focus on different dimensions for classification or efficient management of SO questions and answers. For classification or post quality prediction, textual features of a question [2, 16], characteristics of a code segment [8], social and emotional factors [14], and various key-terms [20] are analyzed. Serva et al. [20] mine negative code examples from Stack Overflow using sentiment analysis where they use a set of key-terms as negativity indicators. However, such indicators might not be sufficient enough to classify a programming related question given that the question generally contains both texts and code segments. Thus, we leverage the human emotions embedded in the text of a question for predicting its quality from Stack Overflow. In this paper, we report an empirical study using 38,920 questions from Stack Overflow, and propose a sentiment metric based machine learning model for identifying low-quality and high-quality questions. For question quality prediction, we combine two categories of features— TF-IDF [19] of the key-terms and sentiment polarity of the sentences in the text from a question. In particular, we consider the frequencies of three types of sentences –*positive*, *negative* and *neutral* from the question texts as sentiment metrics detected by Sentiment140 API.



Fig. 1. Schematic diagram of our study

Our proposed classifier model(s) can classify questions with about 70% precision and 74.2% recall. These are 8% and 2.72% improvements respectively over baseline performance [20]. We also note that sentiment-metrics alone can classify the question from Stack Overflow with about 68.50% precision and about 73.80% recall. Such findings validate our problem statement that human emotions/sentiment embedded in the question text can be a potential indicator of its quality. Thus, we make the following contributions in this paper:

- An exploratory study on how embedded emotions in the text of a question can affect the perception of its overall quality using 38,920 questions from Stack Overflow.
- A question quality prediction model by employing novel sentiment based metrics.

II. METHODOLOGY

Salient feature extraction from the raw data for item classification is a challenging task. In Stack Overflow, each question generally has natural language texts and code segment(s) or code-like elements. Since we focus on embedded emotions in the question texts, we extract the items related to human emotions or sentiments carefully from the questions. Our research methodology comprises of three main steps—(1) Dataset preparation, (2) Exploratory analysis, and (3) Question quality model design. Fig. 1 shows the schematic diagram of these steps. In the following section we describe each of these steps in details.

A. Dataset Preparation

In this research, we wanted to study recent questions from Stack Overflow. We thus collect questions submitted between January, 2014 and January, 2015. We first collect 50K questions using Stack Exchange Data Explorer¹. We found 3,239 down-voted questions, among them only 163 have vote-value less than -4. In order to increase the down-voted samples in our dataset, we collect another 7,235 questions that were posted between January, 2013 and January, 2014, and they have vote value < -4 . We discard the questions with 0 vote or having #words < 5 in the text body from the collected questions. Thus we have a collection of 38,920 questions, 73% of them are up-voted (i.e., $score > 0$) whereas the rest 10,474 (27%) are down-voted (i.e., $score < 0$). Given that concept of quality might be subjective, we consider that a question is of low quality if it is down-voted by the technical crowd of Stack Overflow and vice versa as was also considered by [2, 8, 16, 20]. We then label the low quality and high quality questions using ‘-1’ and ‘+1’

¹<http://data.stackexchange.com/stackoverflow/query/new>

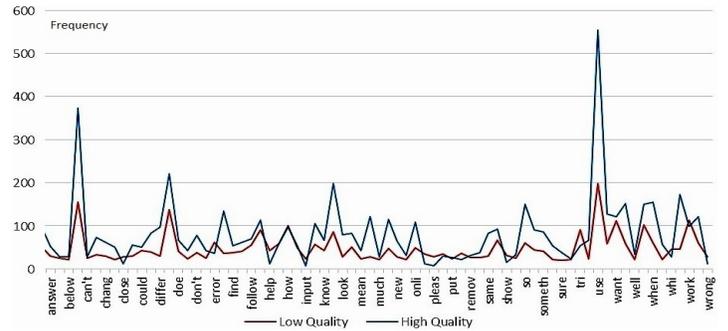


Fig. 2. Base term histograms for low quality and high quality questions from Stack Overflow (the graph is for 76 terms while the diagram producing systems hide most of the terms)

respectively which gives two classes convenient for our SVM and MLP classifier models. The texts from each Stack Overflow question contain various HTML tags including `<code>` tags. These `<code>` tags generally contain code segments or code like elements [18]. Since we focus on natural language texts from a question, we separate the rest content from `<code>` tags, and consider them as the content of interest from the question.

B. Exploratory Study

In order to answer RQ₁, we randomly select 1,000 questions from our dataset containing 500 up-voted questions and 500 down-voted questions. We first perform natural language preprocessing such as stop words (i.e., I, we, to, and, Java...etc.) removal, word splitting and stemming [18], and then determine the term frequency to derive meaningful insights. Fig. 2, and 3 summarize our comparative analysis between different types of questions.

We collect and investigate frequent terms from down-voted questions where we consider a heuristic term frequency of 20 (i.e., to limit the list to the words with greater significance). Since existing studies [20] analyze such questions to extract negative code examples, we select those questions as a starting point for our analysis. This step provides us a collection of 76 terms which we call as *base terms*. We then determine frequencies of the base terms in up-voted questions. Fig. 2 shows the histograms of base terms from the two sets of questions. We note that up-voted questions have relatively greater frequencies compared to the down-voted questions. For example, we see the base terms—‘below’, ‘differ’, and ‘use’ for up-voted questions have the greater frequencies. Mann Whitney U-tests (for up-voted and down-voted questions, the p -value are 0.000060 and 0.00016 which are less than threshold 0.05) also show significant differences in their frequencies. All these findings suggest that there might exist a significant and meaningful difference in the term distribution between up-voted and down-voted questions from Stack Overflow.

We also qualitatively investigate the base terms (i.e., highly frequent words), and notice that some of them reflect human emotions, and are also considered by existing sentiment-based studies [20]. Due to this, it is not enough to apply indicators words for sentiment for the posts.

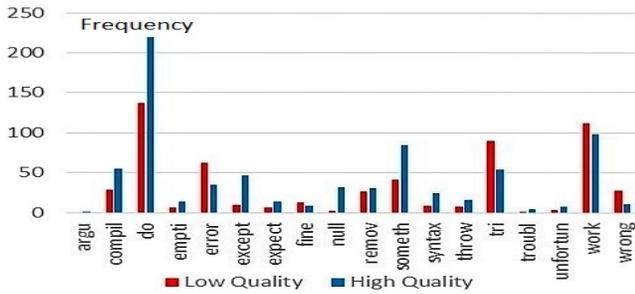


Fig. 3. Histograms of indicator terms from low quality and high quality questions (words are stemmed)

TABLE I

INDICATOR TERMS (TAKEN FROM [20]) AND TOP RANKED NEGATIVE SYNSETS (TAKEN FROM[4])

Indicator Terms		Negative Synsets	
error	expect	abject	unfortunate
wrong	doing	distressing	dispossessed
except	remove	pitiful	tawdry
null	fine	sad	hapless
work	syntax	sorry	miserable
something	try	bad	misfortunate
compile	empty	unfit	pathetic
argue	trouble	unsound	piteous
throw	trying	scrimy	pitiable
unfortunate	getting	shoddy	poor

1) *Key Term Analysis*: Serva et al. [20] propose a list of indicator terms (also called key terms) for extracting low-quality code examples from programming questions and answers of SO. The left column of Table I shows some of their indicator terms. In order to investigate if such key terms can be a distinguishing feature between high quality and low quality questions, we determine their frequencies from our sample questions. Fig. 3 shows the key term histograms for the two sets of our sampled questions. Although the diagrams show moderate frequency differences for the key terms between up-voted and down-coted questions, they are not significant. In particular, Mann-Whitney U tests confirm that the differences are not significant (i.e., $U = 127, p - value = 0.258 > 0.05$). The findings show that while key terms might be effective for negative code example extraction, they are not sufficient enough to separate low quality questions from high quality questions. Despite this, as the test is for 1000 samples and key-terms have moderate histogram differences we consider these terms for further verification in the classification model. In fact, as several studies [4, 21] suggest, human emotions are often captured in sentences or phrases rather than in isolated words.

2) *Sentiment Analysis*: Sentiment analysis identifies the emotions embedded within the texts. Emotions are annotated as- *positive*, *negative* and *neutral*. Over the past years sentiment analysis has been widely adopted, and a popular lexical resource SentiWordNet [4] is used in many research works [21]. In our exploratory study, we first make use of SentiWordNet to contrast between high quality and low quality questions from SO. We collect top-ranked negative synsets from SentiWordNet, and the right column of Table I shows the negative words [4]. We then determine the frequency of these terms in our sampled questions. Fig. 4 shows the histograms for our sampled up-

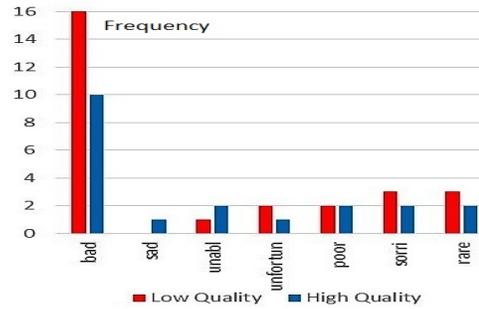


Fig. 4. Histogram of top negative terms (taken from [4]) from 2,000 sampled questions of Stack Overflow

TABLE II

POLARITY OF EXAMPLE QUESTIONS USING *Sentiment140* API

#	ID	Part of Question Text (with vote)	Polarity
1	19393251	This started happening lately, every I use a piece of code it counts [...] I have due in a month because of this stupid problem . {-10}	Negative
2	896532	That aside, I've managed to add Captcha to my comments, and I've learned that customizing the form is a terrible idea [...]. {-3}	Negative
3	8908508	[...] So I will have to separate all this in packages, [...] Is this separation a good thing in your opinion or is this a complete overload? {7}	Neutral
4	8908224	I do not mind keeping the executable but [...] I would love it if the library works just like MySQLdb does. {3}	Positive

voted and down-voted questions. It should be noted that we consider only a tiny fraction (top 10 from the origin list [4]) of the 1,105 synsets as they are well defined by Baccianella et al.[4]. However, the histogram suggests a moderate distribution for those terms. Furthermore, we note that negative words are more frequent in the down-voted (i.e., low quality) questions than the up-voted (i.e., high quality) questions. Existing studies [5, 15] are also found to be aligned with our finding. Bazelli et al. [5] analyze personality traits of the authors from SO questions, and report that the authors of high-voted artifacts (questions, comments, and answers) often express less negative emotions than the authors of down-voted artifacts.

We determine the sentiment polarity of each of the sentences from the example questions using *Sentiment140* API. From Table II, we see that the API provides correct polarities for examples 1, 2 and 4. Unfortunately, the API produced Neutral polarity for sample 3, which should have been Positive. We also perform emotion extraction from the sampled 2000 questions (1000 down-voted and 1000 up-voted), and determine correlation among different attributes (Section II-C1) of the questions to better understand the impact of human emotions. From Table III, we note that positivity in the question is correlated with TF-IDF (term-frequency and inverse-document-frequency) and with overall score of the question. According to our base term analysis, high quality (i.e., up-voted) questions are richer in content, i.e., have greater TF-IDF than low quality (i.e., down-voted) questions. Thus, positive sentiment is

TABLE III
CORRELATION BETWEEN SENTIMENT POLARITY AND QUESTION QUALITY

	NP (Negative)	PP (Positive)	OP (Neutral)
TF-IDF	0.12	0.54	0.35
Question score	0.08	0.15	0.10

positively correlated with question quality as well. On the other hand, we see from Table III that negativity in the question is helpful neither for scoring votes from the crowd nor for improving the content quality of the question.

From the above empirical study, overall, we notice interesting differences between the two types of questions. First, high quality questions are more positive sentimentally than low quality questions, and this positivity also helps them score votes. Second, low quality questions are affected with negativity, and they are also poor in the textual content, i.e., less TF-IDF. Thus, to answer **RQ₁**, the quality of a question is moderately affected by its author’s emotions or sentiments embedded in its natural language texts.

C. The Proposed Question Model

Our exploratory study suggests that high quality and low quality questions from Stack Overflow significantly differ in terms of TF-IDF [19] and human emotions embedded within the texts. That means, such features can be exploited to separate low quality questions from high quality questions. We thus incorporate those features in a question quality model based on machine learning. In this section, we discuss the detailed design of our model, and also attempt to answer **RQ₂**.

1) *Feature Calculation*: Based on the findings from the exploratory study, we select four features for our model– (1) TF-IDF of key/indicator terms (Table I) in the question texts, (2) negative sentence count, (3) positive sentence count and (4) neutral sentence count from a question. Since term distribution differs between high quality and low quality questions according to our exploratory analysis, we determine TF-IDF of a question using the indicator list of [20]. While the TF-IDF is based on the lexical aspect of a question, the remaining three counts are based on emotional signatures within the texts.

Sentiment Polarity Counts: We determine the counts of three types of sentences from a question d , and call them *negative sentence count* (NP_d), *positive sentence count* (PP_d) and *neutral sentence count* (OP_d). In order to determine sentiment polarity, we use Sentiment140 API¹, an implementation of Go et al. [9]. We first isolate each of the sentences (S) from the question texts, determine their polarities using the API, and then accumulate them to generate the three sentence counts:

$$\text{Negative Sentence Count, } NP_d = \sum_{s \in S} I(Pol_s = N_{eg})$$

$$\text{Positive Sentence Count, } PP_d = \sum_{s \in S} I(Pol_s = P_{os})$$

$$\text{Neutral Sentence Count, } OP_d = \sum_{s \in S} I(Pol_s = O_{bj})$$

Here, Pol_s is the polarity indicator of each sentence s , and the identity function $I(Pol_s = N_{eg})$ returns 1 only when

¹<https://cran.r-project.org/src/contrib/Archive/sentiment/>

the polarity Pol_s of s is negative. N_{eg} represents negative polarity, P_{os} means positive polarity, and O_{bj} means neutral polarity. Investigation shows that the up-voted (i.e., high quality) question not only contains meaningful textual content (i.e., higher TF-IDF) but also its content is enriched with positive emotions. On the other hand, the down-voted (i.e., low quality) question not only contains less meaningful content but also the content is affected by negative human emotions.

2) *Question Classification*: Since our dataset contains clearly labeled question samples, we apply supervised learning to our classification task. For classification, we use two machine learning algorithms– Multilayer Perceptron (MLP) and Support Vector Machine (SVM). We train our algorithms using 38,920 question samples from the dataset, and then evaluate the classifiers using 10-fold cross-validation.

Multilayer Perceptron: Multilayer feed forward neural networks are universal approximators and are used to extract patterns or detect trends that are too complex to be noticed [12]. We develop our MLP model using one input layer containing 4 nodes, one hidden layer with 4 nodes and one output layers with 2 nodes. Nodes in the input and output layers are based on the extracted features (i.e., 4) and the labeled classes (i.e., 2). There is no strict rule for selecting the number of nodes in the hidden layer. However, as a rule² of thumb, we use the following formula, and choose four nodes in the hidden layer.

$$\#hidden\ nodes = (\#input\ nodes + \#output\ nodes) \times \frac{2}{3}$$

Although we explored other values, finally we set learning rate=0.1 and number of iterations=500 for the best outcome of our experiments. Since our data might not be linearly separable, we use a logistic function based activation.

Support Vector Machine: Support Vector Machine is often used for detecting binary classes from nonlinear data space. SVM defines support vectors for separating hyperplanes [7]. Since our data might not be linearly separated, we choose Gaussian Radial Basis function [7] as our kernel function for producing non-linear hyperplanes. The value of regularization parameter λ can be arbitrarily adjusted, and through iterative investigations we found that the value 0.05 produces the best result. We run our experiments on WEKA³, and verify our experimental findings for different configurations.

III. EXPERIMENT AND DISCUSSION

We run both classifier models–MLP and SVM– on 38,920 labeled samples where each sample is represented as a vector of four numeric feature values and a class label (i.e., either ‘+1’ or ‘-1’). We apply 10-fold cross-validation for testing the performance of our developed models. For evaluation and validation, we use two classical performance metrics– precision and recall. Table IV and VI summarize our experimental results using those metrics.

Table IV shows how our models perform with different sets of features considered with noisy data. We first run experiments

²<http://stackoverflow.com/questions/10565868>

³<http://www.cs.waikato.ac.nz/ml/weka/>

TABLE IV
EXPERIMENTAL RESULTS (WITH NOISE)

	Features	SVM		MLP		Serva et al. [20]	
		Precision	Recall	Precision	Recall	Precision	Recall
Baseline Method	$\{S(t)\}$	–	–	–	–	61.61%	71.48%
	$\{TF - IDF, NP, PP, OP\}$	63.40 %	72.00%	69.60%	74.20 %	–	–
Proposed Method	$\{TF - IDF, NP, PP\}$	63.10 %	72.30 %	69.40%	74.10%	–	–
	$\{NP, PP\}$	67.60 %	73.80%	68.50%	73.80%	–	–

TABLE V
RESULTS FOR BALANCED DATASET(WITH NOISE)

	Features	SVM		MLP		Serva et al. [20]	
		Precision	Recall	Precision	Recall	Precision	Recall
Baseline Method	$\{S(t)\}$	–	–	–	–	66.44%	74.21%
	$\{TF - IDF, NP, PP, OP\}$	82.7 %	82.3%	82.6%	82.1%	–	–
Proposed Method	$\{NP, PP, OP\}$	82.2 %	82.00%	82.8%	82.6%	–	–

TABLE VI
EXPERIMENTAL RESULTS FOR 700 RANDOM QUESTIONS (WITHOUT NOISE)

Features	SVM		MLP	
	Precision	Recall	Precision	Recall
$\{TF-IDF, NP, PP, OP\}$	67.00 %	66.10%	69.00 %	68.90 %
$\{NP, PP\}$	66.50 %	65.40 %	68.50 %	68.80%

with indicator term based technique of Serva et al. on our dataset, and found 61.61% precision and 71.48% recall. We consider them as the baseline performance.

Then we perform experiments with our MLP and SVM models, and found MLP performing better than SVM. They show about 8% improvement in the precision and 3% improvement in the recall over the baseline performance. Since we are interested to investigate how emotional components (i.e., sentiments) in the texts affect the quality of a question, we filter out additional features gradually and collect the results. We first discard the neutral sentence count (*OP*), and notice very trivial change in the performance. This suggests that this feature has a very low predictability power for question quality. We then discard TF-IDF from our models, and notice that the performance is still almost the same. For example, with the two sentiment based features—negative sentence count (*NP*) and positive sentence count (*PP*), our MLP classifier provides a 68.50% precision and 73.80% recall which are promising. Although the accuracy is not too high, these findings clearly pose the sentiment based metrics as promising candidate features for quality prediction of Stack Overflow questions.

As our dataset contained unexpected noise that might be affecting our findings, we choose a random subset of 1,000 questions from the dataset, and discard the questions containing items other than pure texts—*equation*, *email address*, *code segment*, *hyper-links*, and *configuration information*. This leads us to a collection of 700 samples containing 400 up-voted questions and 300 down-voted questions, and our classifier provides 69.00% precision and 68.50% recall (Table VI) with these questions (and same configuration of the model). These are very close to our previously reported findings in Table IV.

Finally, we prepare balanced dataset (51 : 49) by separating 11,000 top up-voted questions and 10,474 down-voted ques-

tions, and employ our model on this. In this case, the outcome is significantly improved; precision rates for MLP and SVM are 82.6 % and 82.2 % respectively with the sentiment features only (Table V). Thus, to answer **RQ₂**, machine learning model can effectively predict the quality and classify the questions based on sentiment metrics. Such findings also strengthen our **RQ₁** that embedded human emotions in the question texts can really affect the quality of a question.

IV. RELATED WORK

Mining of programming Q & A sites containing millions of technical questions, answers and comments are becoming more and more popular among the research community. Arai and Handayani [2] present a general model to predict quality of information from Yahoo! answers by using answer features (AF) as metrics (i.e., non-textual information). They use only 815 training samples and 302 test samples. Most of the metrics they use are calculated during post-submission phase of a question, and are heavily dependent on the time spent. Ponzanelli et al. [16] use SO metrics, Readability metrics, and Popularity metrics for predicting the quality of technical questions where they employ decision tree (DT) as the predictive model. SO metrics and Popularity metrics are post-submission metrics, and are dependent on questions age. Ponzanelli et al and colleagues [17] also suggest a linear quality function (QF) to submit the low quality questions into review queue. In a recent study, Duijn et al. [8] focus on code only information for classification of posts. One of the features they selected is the length of the code segment. However, from the manual investigation of posts it is observed that there exist numerous code segments which are positive but have fewer lines of code, even in some cases one line of code. Moreover, many questions do not contain code-segment, thus the quality is only dependent on the text of those questions.

Several studies are available to detect human emotions [1, 9, 21] in the informal texts from social networking website. However, corpora of Q & A sites mainly contain unstructured technical texts and mining them is challenging due to emoticons, slang, misspelled words, hash-tags, links, e-mail, equation, and even code-segments within a sentence [15]. Recently,

Novielli et al. [15] conduct an empirical study on 400 top scored SO posts (questions, answers, and comments) using SentiStrength tool, and identify several challenges of sentiment analysis in technology domain. Despite limitations of affect (i.e., human emotion) extraction, several researchers focus on software developers emotions and behavior in software artifacts [5, 11, 13]. Murgia et al. [13] investigate emotional information of the developers in the software development. They analyze issue reports from issue trackers (Apache Jira), and categorize the found emotions into six groups. Bazelli et al. [5] study SO questions and answers to explore the personality traits of the authors, and report that authors of high-voted posts reveal notably less negative emotions than the authors of down-voted posts. Serva et al. [20] investigate 240 posts, and extract key-terms having negative impact on question quality. Although, these terms are claimed to be negative, our study suggests their little power in distinguishing lower quality questions from higher quality questions (Section II-B). In a related work, Novielli et al. [14] argue about the impact of emotions upon the quality of SO posts. However, their exists no implementation of their work yet. In addition, there are other sentiment extraction related studies [10, 22] which motivate us to dig down more deeply about the problem. To summarize, most of the SO question’s quality analysis use features (SO metrics and popularity metrics) which are based on the age of the questions along with textual metrics whereas our proposed models do not depend on them. We consider the sentiment polarity of each of the sentences from the question text as features for designing the model. Thus, our proposed approach is more feasible for quality prediction during question posting. Moreover, our experiment is also a feasibility evaluation of Sentiment140 API to the detection of human emotions from technical discussion text.

V. THREATS TO VALIDITY

Although we conduct our study and analyze our results carefully, there exist several threats to the validity of our findings. First, programming questions contain various items beyond natural language texts such as equations, emails, symbols, hyper-links and code-segments which can pose as noise, and the reported results might be affected. In order to handle that threat, we experiment with 700 randomly selected questions after removing noise, and collect the results. Interestingly, we note that the results do not vary significantly from our original performance.

Second, our dataset is inherently skewed since there is a bit imbalance in the categories of questions (73% up-voted vs 27% down-voted). We did our experiment with what we get randomly. Thus the classifiers also do not perform equally for both types of questions. However, experiment with balanced dataset (51% up-voted and 49% down-voted), shows better performance (Table V) for our model, and thus that was not really a threat.

Third, we consider positive-scored questions as of high quality and negative-scored questions as of low quality as suggested by relevant literature [2, 8, 16, 20]. That means, we

exploit crowd-sourced knowledge to develop the oracle for our classifiers, and still we cannot guarantee that all positive-scored questions contain high quality content.

VI. CONCLUSION AND FUTURE WORK

Programming questions in Stack Overflow are rapidly increasing, and maintaining their content quality is a major concern. In this paper, we report an exploratory study and experimental outcome of a proposed model using 38,920 questions from Stack Overflow where we investigate the impact of author’s emotions on the quality of a question. We select four features including three sentiment based features for separating low quality questions from high quality questions, and perform the classification using two popular machine learning algorithms—MLP and SVM. Our best model, MLP provides a precision of 70% and a recall of 74% which are promising according to relevant literature. We also clearly demonstrate that human emotions embedded in the question texts have significant impact on the quality of the questions. In future, we plan to explore the scope of sentiment aspects in various classification, filtration and management of StackOverflow questions.

REFERENCES

- [1] A. Agarwal, B. Xie, I. Vovsha, O. Rambow, and R. Passonneau. Sentiment analysis of twitter data. In *Proc. LSM*, pages 30–38, 2011.
- [2] K. Arai and A. N. Handayani. Predicting quality of answer in collaborative q and a community. *Intl. J. of Adv. Research in AI*, pages 21–25, 2013.
- [3] M. Asaduzzaman, A. S. Mashiyat, C. K. Roy, and K. A. Schneider. Answering questions about unanswered questions of stack overflow. In *Proc. MSR*, pages 97–100, 2013.
- [4] S. Baccianella, A. Esuli, and F. Sebastiani. Sentiwordnet 3.0: An enhanced lexical resource for sentiment analysis and opinion mining. In *Proc. LREC*, pages 2200–2204, 2010.
- [5] B. Bazelli, A. Hindle, and E. Stroulia. On the personality traits of stackoverflow users. In *Proc. ICSM*, pages 460–463, 2013.
- [6] D. Correa and A. Sureka. Chaff from the wheat: Characterization and modeling of deleted questions on stack overflow. In *Proc. WWW*, pages 631–642, 2014.
- [7] C. Cortes and V. Vapnik. Support-vector networks. *Machine Learning*, pages 273–297, 1995.
- [8] M. Duijn, A. Kucera, and A. Bacchelli. Quality questions need quality code: classifying code fragments on stackoverflow. In *Proc. MSR*, pages 410–413, 2015.
- [9] A. Go, R. Bhayani, and L. Huang. *Twitter sentiment classification using distant supervision*. Stanford University, Tech. Rep, 2009.
- [10] E. Guzman, D. Azócar, and Y. Li. Sentiment analysis of commit comments in github: An empirical study. In *Proc. MSR*, pages 352–355, 2014.
- [11] E. Guzman and B. Bruegge. Towards emotional awareness in software development teams. In *Proc. FSE*, pages 671–674, 2013.
- [12] K. Hornik, M. Stinchcombe, and H. White. Multilayer feedforward networks are universal approximators. *Neural Netw.*, pages 359–366, 1989.
- [13] A. Murgia, P. Tourani, B. Adams, and M. Ortu. Do developers feel emotions? an exploratory analysis of emotions in software artifacts. In *Proc. MSR*, pages 262–271, 2014.
- [14] N. Novielli, F. Calefato, and F. Lanubile. Towards discovering the role of emotions in stackoverflow. In *Proc. SSE*, pages 33–36, 2014.
- [15] N. Novielli, F. Calefato, and F. Lanubile. The challenges of sentiment detection in the social programmer ecosystem. In *Proc. SSE*, pages 33–40, 2015.
- [16] L. Ponzanelli, A. Mocchi, A. Bacchelli, and M. Lanza. Understanding and classifying the quality of technical forum. In *Proc. QSIC*, pages 343–352, 2014.
- [17] L. Ponzanelli, A. Mocchi, A. Bacchelli, M. Lanza, and D. Fullerton. Improving low quality stack overflow post detection. In *Proc. ICSME*, pages 541–544, 2014.
- [18] M. M. Rahman and C. K. Roy. An insight into the unresolved questions at stack overflow. In *Proc. MSR*, pages 426–429, 2015.
- [19] S. Robertson. Understanding inverse document frequency: On theoretical arguments for idf. *Journal of Documentation*, 60, 2004.
- [20] R. Serva, Z. Senzer, L. Pollock, and k Vijay-Shanker. Automatically mining negative code examples from software developer Q & A forums. In *Proc. SoftMine ASE*, pages 410–413, 2015.
- [21] C. Strapparava and R. Mihalcea. Learning to identify emotions in text. In *Proc. ACM Applied computing*, pages 1556–1560, 2008.
- [22] Y. Zhang and D. Hou. Extracting problematic api features from forum discussions. In *Proc. ICPC*, pages 142 – 151, 2013.

Multi-Objective Optimization for Software Testing Effort Estimation

Solomon Mensah¹, Jacky Keung¹, Kwabena Ebo Bennin¹ and Michael Franklin Bosu²

¹Department of Computer Science, City University of Hong Kong, Hong Kong, China
{smensah2-c, kebennin2-c}@my.cityu.edu.hk, Jacky.Keung@cityu.edu.hk

²Centre for Business, Information Technology and Enterprise
Wintec, Hamilton, New Zealand
michael.bosu@wintec.ac.nz

Abstract— Software Testing Effort (STE), which contributes about 25-40% of the total development effort, plays a significant role in software development. In addressing the issues faced by companies in finding relevant datasets for STE estimation modeling prior to development, cross-company modeling could be leveraged. The study aims at assessing the effectiveness of cross-company (CC) and within-company (WC) projects in STE estimation. A robust multi-objective Mixed-Integer Linear Programming (MILP) optimization framework for the selection of CC and WC projects was constructed and estimation of STE was done using Deep Neural Networks. Results from our study indicate that the application of the MILP framework yielded similar results for both WC and CC modeling. The modeling framework will serve as a foundation to assist in STE estimation prior to the development of new a software project.

Keywords- Software Testing Effort; Cross-Company; Within-Company; Optimization; Deep Neural Networks

I. INTRODUCTION

Software Testing Effort (STE) is one of the critical phases in software development as it is estimated to contribute about 25-40% [1] of the total development cost. STE is the effort in relation to duration, resources and source code for testing the software before deployment to the client or market [1]. The acquisition of relevant data for effort estimation has been a focus for prior work [2] concerning the effective use of within-company (WC) and cross-company (CC) datasets for predictive modeling. Researchers and practitioners who lack the relevant local data rely on data imported from other repositories or companies. The challenge is that this approach might not always be useful for WC estimation. In order for WC estimation to benefit from CC datasets, the selection of an optimal “mix” of project datasets and features is required for efficient estimation.

The WC and CC modeling have been considered in the domains of software development effort estimation and defect prediction [2][4]. To the best of our knowledge, this is the first study to employ the use of CC projects in the estimation of STE. A study by Burak et al. [4] indicates that models that used CC datasets for defect prediction yielded similar performance to WC models when normalization and Nearest Neighbor filtering techniques were applied to the datasets.

In this study, we incorporated a z-score normalization

technique into our proposed project selection approach after it proved superior to log transformation and box cox transformation.

The two research questions addressed by this study are as follows: **RQ1**. Do models constructed from CC projects yield similar STE results to models from WC projects with/without normalization? **RQ2**. Does STE estimated from “prior” features yield similar results to STE estimated from “posterior” features?

In order to minimize the cost (effort) associated with project and feature selection in STE estimation, this study proposes a multi-objective Mixed-Integer Linear Programming (MILP) optimization framework. The selection of an optimal “mix” of N^* projects with their respective f^* features from m companies to estimate STE is subjected to the following constraints: 1) allocation of projects constraint; 2) allocation of features constraint; 3) size of project constraint.

The rest of the paper is organized as follows: Section II describes the related work. Section III presents the methodology. Section IV describes the results from the study. Finally, Section V presents the conclusion and future work.

II. RELATED WORK

Bareja and Singhal [1] studied various effort estimation techniques in order to minimize STE. They realized that machine learning and data mining testing techniques adopted by developers assist in the reduction of effort expended in the software development process. A study by Turhan et al. [4] investigated how cross-company (CC) projects can be used for defect prediction modeling. They found that within-company (WC) modeling performed better than CC modeling. In the field of effort estimation, Kocaguneli and Menzies [2] in the quest for finding relevant dataset for effort estimation concluded that there is little significant difference in the use of CC or WC datasets for modeling. Ansari et al. [7] proposed a regression test case optimization approach to assist in the reduction of error cost and testing time to achieve a more quality software product. Their test selection approach makes use of prioritized test cases for testing riskier components of the product in order to enhance system reliability and stability.

This study differs from previous works since to the best of our knowledge this is the first study to apply a multi-objective MILP for project and feature selection for STE estimation.

III. METHODOLOGY

A. Datasets

For the purpose of this study, 45 *PHP* projects from 5 software development companies were used. We present the projects with the means of their respective sampled features - Function Points (FP), Development Duration (DD) in months, Lines of Codes (LOC) and STE in person-months in Table I. Other features from the projects are Number of Development Personnel (DP), Test Cases (TC), Project Cost (PC), number of defects, user stories and marketers. For confidentiality reasons, we denote the names of the companies with X_j and their respective project names as X_{ij} .

TABLE I. DESCRIPTIVE SUMMARY OF COMPANY PROJECTS

Company	No. of Projects	Mean			
		FP	DD	LOC	STE
X_1	15	47.2	13.7	14589	151.7
X_2	7	44.4	8.4	3942	25.0
X_3	4	301.0	16.0	719786	123.3
X_4	4	58.8	4.8	14654	18.6
X_5	15	238.0	7.3	65543	25.5

B. Within-Company and Cross-Company Modeling

Given $\{X_1, X_2, \dots, X_m\}$ denoting a set of m companies and each X_j consisting of a set of $\{X_{1j}, X_{2j}, \dots, X_{mj}\}$ projects, then we formulate a within-company (WC) project modeling as a combination of selected X_{ij} projects from a single X_j company. Similarly, cross-company (CC) project modeling as a combination of projects from a number of companies. In each case of the modeling, the selected projects by the MILP were used for constructing the predictive model for estimating the STE of a target project.

Based on findings by Turhan et al. [4], we also found the need of preprocessing datasets prior to model construction. We applied a 3-step preprocessing approach. 1) Data Cleaning: In order to assess if the datasets were normally distributed, we made use of a graphical analytical technique namely Quantile-Quantile (QQ) plot. It should be noted that, no missing values were observed in the datasets used. 2) Data Transformation: Because of variations within the datasets, we used the z -score normalization to transform the data points into specific range in order to leverage the outweighing discrepancies. 3) Data Reduction: The MILP framework did not select three projects and this was due to the weak correlation between the independent features and the dependent feature (STE).

C. Proposed MILP Algorithm

The MILP optimization framework is formulated for the project and feature selection. Deep Neural Network (DNN) is incorporated into the framework for estimating the STE. A 7-step procedure of the proposed framework is presented below.

Step 1: Preprocess project datasets.

Step 2: Given m companies each with an archival set of projects, construct the MILP for the project selection.

Step 3: For every selected optimal set of X_i^* projects from each j^{th} company, compute the multivariate Spearman rank correlation between the respective independent features and the dependent feature (STE).

Step 4: Select the projects based on the project selection optimization method and based on the higher correlation values in a non-increasing order until the optimal number of projects for every j^{th} company is achieved.

Step 5: Combine the selected projects from each j^{th} company. For the given project datasets applied, it was assumed that each project has similar set of features for estimating STE.

Step 6: Select the features based on the feature selection optimization method.

Step 7: Construct both within-company and cross-company STE models in the domain of DNN and evaluate the performance with MmRE, a robust metric free from outliers [9].

D. Optimization Model Formulation

A generalized optimization model formulation specifically MILP is composed of three main components namely; the objective function, the constraints and decision variable bounds. The optimization model minimizes or maximizes an objective function subject to certain constraints, which can be in the form of equality or inequality constraints. The decision variable (normally independent) is said to be discrete if it can assume a finite set values and continuous if it can assume values within a specified interval. Optimization problems with both discrete and continuous decision variables are said to be MILP. Based on Mridul and Rana [6] categorization of software size into functional and technical size, we constructed the generalized objective function for the MILP incorporating the functional size in the form of FP. Thus, since our ultimate aim is to estimate STE prior to development, we eliminated LOC which is a technical size from developers' perspective and made use of FP from users' perspective together with other prior input features - estimated project duration, estimated project cost and number of development personnel. FP is defined as the sum of five components – external inputs, external outputs, external query, external interface files and internal logical files with their respective weights [6]. Posterior features from the projects include LOC, coding time, test cases and number of defects.

In the formulation of the objective functions, we define a probability weight function, α_{ij} for each i^{th} project from the j^{th} company as shown in (1).

$$\alpha_{ij} = \frac{(\sum_{j=1}^p f_j)_i}{\sum_{i=1}^n (\sum_{j=1}^p f_j)_i} \quad (1)$$

where p denotes the total number of f_j features for every project and n denotes the total number of features from all companies. Each of the probability weights is multiplied by the respective prior input features. We incorporated two unique identifiers, λ_{ij} to uniquely label every i^{th} project from the j^{th} company and β_j to uniquely identify each of the companies as shown in (3) and (4).

Objective Function

In this study, we formulated two main objective functions for the MILP problem – WC and CC objective functions. Each objective function is further categorized for project and feature selection. All objective functions are minimization of project cost and feature selection problems from the respective companies.

A mathematical model based on Parkinson's Law [5] is formulated in (2) for computing the STE for each unsupervised project prior to the setting up of the MILP model. Here, unsupervised project refers to dataset without STE.

$$STE = (\sum_{i=1}^t DD \times \sum_{j=1}^t DP) \times TP\% \quad (2)$$

where DD = Development Duration in months; DP = Development Personnel; TP = Testing Proportion. For

consistency in the computation of the STE for each unsupervised project dataset, we chose a threshold of 40% for TP [1]. DP and DD are parameters denoting the total number of development personnel and duration for requirement gathering, design and testing respectively.

WC Objective Function for Project Selection

We first define the cost function, $Cost^{WCP}$ in relation to project selection for WC in (3) as a cost minimization involving the following design variables – DD, DP, Function Points (FP), Test Cases (TC), Project Cost (PC), Probability Weight Function (α), Project Identification Code (λ)

$$Cost^{WCP} = \alpha_{ij} \left(\sum_t \sum_n DP_n DD_t + \sum_u TC_u + \sum_i \sum_j FP_{ij} + \sum_c PC_c \right) + \sum_i \sum_j \lambda_{ij} \quad (3)$$

CC Objective Function for Project Selection

The objective cost function, $Cost^{CCP}$ for the CC project selection is defined in (4).

$$Cost^{CCP} = \alpha_{ij} \left(\left(\sum_t \sum_n DP_n DD_t + \sum_u TC_u + \sum_i \sum_j FP_{ij} + \sum_c PC_c \right) + \sum_i \sum_j \lambda_{ij} \right) + \sum_j \beta_j \quad (4)$$

WC Objective Function for Feature Selection

We define an expression for the objective function, $Cost^{WCF}$ for the optimal subset of features for WC in (5). The $Cost^{WCF}$ function incorporates the Akaike Information Criteria (AIC) for the optimal selection of features for the cross modeling process.

$$Cost^{WCF} = \arg \left\{ \min_{r, \delta, k \in \mathbb{R}} \{ r \log(\delta^2) + 2k \} \right\} \lambda_{ij} \quad (5)$$

where r = number of records in the cross projects, δ^2 = mean squared error, k = number of estimated parameters in AIC. Here, we consider STE as the dependent variable and independent variables are the rest of the features to be selected. We therefore constructed an ordinary least squares regression and used AIC incorporating both forward and backward selection with the aim of selecting the optimal subset of features. AIC is very good at handling much more complex models and achieves a better bias-variance tradeoff [10].

CC Objective Function for Feature Selection

The objective function, $Cost^{CCF}$ for the optimal subset of features for the CC approach is defined in (6).

$$Cost^{CCF} = \arg \left\{ \min_{r, \delta, k \in \mathbb{R}} \{ r \log(\delta^2) + 2k \} \right\} \beta_j \quad (6)$$

Constraints

We considered three constraints namely; allocation of projects, allocation of features and size of projects constraints.

Allocation of Projects Constraint

In order to minimize cost, we considered an inequality constraint for the allocation of projects defined as $\sum_i \sum_j X_{ij} < Y_N$ whereby not all the company projects can be selected. The variable X_{ij} denotes the i^{th} project selected from the j^{th} company. Y_N denotes the total number of N projects from the companies.

Secondly, we considered the inequality constraint, $\sum_i X_i \leq X_j^* < Y_N$ for the selection of projects from within-company. X_j^* denotes the total number of projects from a given j^{th} company. We assumed that at the worst case scenario, all projects can be selected from most of the companies but not all companies.

Allocation of Features Constraint

We define an inequality constraint for the allocation of features in terms of Variance Inflation Factor (VIF) to deal with the multicollinearity issues among the predictor (independent) features. After experimental fine tuning of the

parameters, we considered multicollinearity as an issue if VIF is more than a specified threshold, $k = 10$ in each AIC model. Hence, for optimal predictor features to be obtained we needed correlation between predictor features to be very minimal.

An adjusted R^2 of 0.7 was chosen after parameter fine tuning. Hence, for optimal feature subset, we needed at least 70% of the total variation in STE to be explained by the predictor features.

Lastly, we considered an inequality constraint in the form of $\sum_i \sum_j \sum_p f_{ijp} < f_N$ in order not to select all features from the total projects. Thus, in minimizing cost, we made the assumption that, there exist a subset of features (f_{ijp}) that will equally play a significant role in estimating STE as compared to all features (f_N). f_{ijp} denotes the selected p^{th} feature from the i^{th} project selected from the j^{th} company.

Size of Projects Constraint

We define two inequality constraints, $\sum_i \sum_j FP_{ij} \leq FP_N$ and $\sum_i \sum_j \sum_p FP_{ijp} < FP_N$ for the project sizes in relation to FP [6]. Here, we considered the sum of FP in the selected projects in a given j^{th} company to be at most equal to the total number of FP in all projects from that j^{th} company. On the other hand, the sum of FP from the selected projects was considered to be strictly less than the total FP in all the projects from the selected companies.

E. Deep Neural Networks (DNN) Model

In the estimation of STE for the selected projects using their respective prior and posterior features, we made use of DNN. DNN was considered for the within-company (WC) and cross-company (CC) modeling approach since it makes use of multiple layers to automatically learn from a set of features and gives better predictive results [8]. After series of fine tuning of network architecture parameters, we considered the DNN for the STE estimation to be best at 3 hidden layers with 5, 2 and 1 neuron(s) respectively and an output layer with a single neuron. The Levenberg-Marquardt backpropagation optimization training function was used to update the weights and the hyperbolic tangent activation function was used in each of the neurons for giving the respective outputs. We employed the k -fold cross validation approach for setting up the DNN model. The formation of the training set and test set differ slightly for the CC and WC projects. For CC modeling, we used all projects from four of the five companies to form the training set whilst the fifth company project formed the test set. This is repeated in a leave-one-out (LOO) cross validation manner till all projects from the companies were part of the training set and test set respectively. For WC modeling, we used all but one project from a single company to form the training set whilst the remaining project formed the test set using the LOO method similar to the CC modeling. We then used the Median Mean Relative Error (MdMRE) [9] in evaluating the DNN model. Experiment was conducted in MATLAB toolkit (version R2014b).

IV. RESULTS AND DISCUSSION

A. Project and Feature Selection by MILP

After applying the MILP to the project datasets, a subset of 42 projects and five features were selected. These features were project duration in months (PD), number of development personnel (DP), number of test cases (TC), number of function points (FP) and the cost of each project (PC) in USD.

RQ1: Do models constructed from CC projects yield similar STE results to models from WC projects with/without normalization?

In order to compare the evaluation performance of STE estimated from both approaches, we used the MdmRE accuracy measure [9] and presented results in Tables II-III. The results in relation to normalized and un-normalized datasets are presented using the win/tie/loss metric. The win/tie/loss metric enabled us to make an evaluation and comparative performance using MdmRE for the within-company (WC) and cross-company (CC) modeling [2]. For example, in Table II, in relation to MdmRE evaluation, we realized that WC and CC modeling with z -score normalization yielded similar predictive results in 3 cases (that is 3 ties in the 2nd, 3rd and 5th cases where one project from X_4 , X_3 and X_1 were used for testing respectively). Without the normalization technique, we realized that WC dominated in estimating STE (Table III). Thus, results from our optimization selection framework and DNN estimation modeling approach reveal that, models constructed from WC and CC yield approximately similar STE results when datasets were subjected to the z -score normalization technique.

TABLE II. MDMRE EVALUATION (NORMALIZED)

CC Train set	WC Train set	Test set	Win	Tie	Loss
$X_{i1}, X_{i2}, X_{i3}, X_{i4}$	$X_{i-1,5}$	LOO	WC		CC
$X_{i1}, X_{i2}, X_{i3}, X_{i5}$	$X_{i-1,4}$	LOO		✓	
$X_{i1}, X_{i2}, X_{i4}, X_{i5}$	$X_{i-1,3}$	LOO		✓	
$X_{i1}, X_{i3}, X_{i4}, X_{i5}$	$X_{i-1,2}$	LOO	WC		CC
$X_{i2}, X_{i3}, X_{i4}, X_{i5}$	$X_{i-1,1}$	LOO		✓	

TABLE III. MDMRE EVALUATION (UN-NORMALIZED)

CC Train set	WC Train set	Test set	Win	Tie	Loss
$X_{i1}, X_{i2}, X_{i3}, X_{i4}$	$X_{i-1,5}$	LOO	WC		CC
$X_{i1}, X_{i2}, X_{i3}, X_{i5}$	$X_{i-1,4}$	LOO		✓	
$X_{i1}, X_{i2}, X_{i4}, X_{i5}$	$X_{i-1,3}$	LOO	WC		CC
$X_{i1}, X_{i3}, X_{i4}, X_{i5}$	$X_{i-1,2}$	LOO	WC		CC
$X_{i2}, X_{i3}, X_{i4}, X_{i5}$	$X_{i-1,1}$	LOO		✓	

LOO – Leave one out; WC – within company; CC – cross company

RQ2: Does STE estimated from “prior” features yield similar results to STE estimated from “posterior” features?

Here, we compared performance of STE estimation from both “prior” and “posterior” features with respect to WC and CC modeling. Due to space limitation, the MdmRE evaluation of results are presented in Tables IV-V available in the link¹. In relation to cross-company (CC) modeling approach, the average MdmRE value was 79.7%. In relation to within-company (WC), the average MdmRE value was 82.0%. Result from Table IV shows that, in relation to the application of the normalization technique, STE estimated from CC modeling yielded similar results for both prior and posterior features. We further confirmed this result using Friedman’s test statistic [3] which yielded p -values of 0.4240 and 0.0597 at 5% significance level for the prior and posterior features respectively. This indicates that there is no statistical difference in the STE estimation results from the prior and posterior features. In the WC modeling, prior features

dominated best in the STE estimation as illustrated in Table V.

This means that, our proposed multi-objective MILP optimization selection framework can select optimal prior features which yield similar STE results as compared to posterior features provided a z -score normalization is applied.

V. CONCLUSION AND FUTURE WORK

A Mixed-Integer Linear Programming (MILP) optimization framework has been proposed for the selection of desirable number of projects with their respective features from within-company and cross-company projects. The five input features selected by the MILP for Software Testing Effort (STE) estimation prior to development are Project Duration, Development Personnel, Test Cases, Function Points and Project Cost. We subjected the selected projects and features to train a Deep Neural Network (DNN) model for estimating STE using the k -fold cross validation approach. Results show that the DNN model for estimating STE from cross-company projects yielded similar results to within-company projects provided that the z -score normalization method was applied.

Going forward, we intend to incorporate a filter in our MILP framework to further improve the CC project and feature selection approach for STE estimation.

ACKNOWLEDGEMENT

This research is supported by the City University of Hong Kong research funds (Project No. 7200354, 7004222, 7004474).

REFERENCES

- [1] K. Bareja and A. Singhal. "A Review of Estimation Techniques to Reduce Testing Efforts in Software Development." *Advanced Computing & Communication Technologies (ACCT), 2015 Fifth International Conference on.* IEEE, 2015.
- [2] E. Kocaguneli and T. Menzies. "How to find relevant data for effort estimation?." *Empirical Software Engineering and Measurement (ESEM), 2011 International Symposium on.* IEEE, 2011.
- [3] M. Friedman. "The use of ranks to avoid the assumption of normality implicit in the analysis of variance." *Journal of the american statistical association* 32.200 (1937): 675-701.
- [4] B. Turhan, T. Menzies, A. B. Bener and J. D. Stefano. "On the relative value of cross-company and within-company data for defect prediction." *Empirical Software Engineering* 14.5 (2009): 540-578.
- [5] C. N. Parkinson. *Parkinson's law, and other studies in administration.* Vol. 24. Boston: Houghton Mifflin, 1957.
- [6] B. Mridul and A. Rana. "Impact of Size and Productivity on Testing and Rework Efforts for Web-based Development Projects." *ACM SIGSOFT Software Engineering Notes* 40.2 (2015): 1-4.
- [7] A. S. A. Ansari, K. K. Devadkar and P. Gharpure. "Optimization of test suite-test case in regression test." *Computational Intelligence and Computing Research (ICCIC), 2013 IEEE International Conference on.* IEEE, 2013.
- [8] K. Sangwook, M. Lee and J. Shen. "A novel deep learning by combining discriminative model with generative model." *Neural Networks (IJCNN), 2015 International Joint Conference on.* IEEE, 2015.
- [9] T. Foss, E. Stensrud, B. Kitchenham and I. Myrvtveit. "A simulation study of the model evaluation criterion MMRE." *Software Engineering, IEEE Transactions on* 29.11 (2003): 985-995.
- [10] D. Ruppert. *Statistics and finance: An introduction.* Springer Science & Business Media, 2004.

¹ MdmRE evaluation results - <http://tinyurl.com/WinTieLoss>

Effectiveness of Human Error Taxonomy during Requirements Inspection: An Empirical Investigation

Vaibhav Anu, Gursimran Walia
Department of Computer Science
North Dakota State University
Fargo, USA
vaibhav.anu@ndsu.edu,
gursimran.walia@ndsu.edu

Wenhua Hu, Jeffrey C. Carver
Department of Computer Science
University Of Alabama
Tuscaloosa, USA
wh10@crimson.ua.edu,
carver@crimson.ua.edu

Gary Bradshaw
Department of Psychology
Mississippi State University
Mississippi State, USA
glb2@psychology.msstate.edu

Abstract—Software inspections are an effective method for achieving high quality software. We hypothesize that inspections focused on identifying errors (i.e., root cause of faults) are better at finding requirements faults when compared to inspection methods that rely on checklists created using lessons-learned from historical fault-data. Our previous work verified that, error based inspections guided by an initial requirements errors taxonomy (RET) performed significantly better than standard fault-based inspections. However, RET lacked an underlying human information processing model grounded in Cognitive Psychology research. The current research reports results from a systematic literature review (SLR) of Software Engineering and Cognitive Science literature - Human Error Taxonomy (HET) that contains requirements phase human errors. The major contribution of this paper is a report of control group study that compared the fault detection effectiveness and usefulness of HET with the previously validated RET. Results of this study show that subjects using HET were not only more effective at detecting faults, but they found faults faster. Post-hoc analysis of HET also revealed meaningful insights into the most commonly occurring human errors at different points during requirements development. The results provide motivation and feedback for further refining HET and creating formal inspection tools based on HET.

Keywords-human error; requirements inspection; taxonomy; empirical study

I. INTRODUCTION

Software engineers spend around 80% of total development time during testing and debugging [13], a majority of which is spent on fixing faults that were committed during the early phases (e.g., requirements development). To address this problem, project managers focus on *software inspections* (i.e., reviewing requirements and design documents to identify faults) when they are easiest and cheapest to find and fix. Empirical evidence reports that, inspections at multiple organizations (e.g., IBM, TRW, ICL, Cardiac pacemakers) have led to significant improvements in quality, reduction in fault rate and delivering products within allocated time and cost [14].

In spite of this wide spread success, inspection techniques focus developers' attention only on different type of *faults* (e.g., missing or incorrect functionality) recorded in software artifacts. As a result, they cannot help inspectors detect all the problems without understanding underlying *errors* (i.e., source of faults). Therefore, we believe that, an inspection process that can help developers' focus on the cause of the problems (as opposed to

just the symptoms of errors - *faults*) will be a significant improvement over standard fault-checklist based inspections. To that end, our prior research [6] has gathered initial evidence to show that structured *error* information help inspectors detect significantly large number of faults (that are otherwise overlooked or left undetected) as compared to *fault-checklist* (FC) based inspections. Even when comparing against the most advanced fault inspection technique (*Perspective based Reading* - PBR [6]), error based inspection performed better. A brief summary of prior research and motivation for the current research is provided in next paragraphs.

Prior Research: To help evaluate the feasibility of an error-based inspection, Walia and Carver (co-authors), created a preliminary classification of requirement errors by classifying errors reported in published literature into three main error types: *People, Process* and *Documentation* Errors. Next, a series of controlled experiments [3-5] validated that, an error-inspection (guided with RET) significantly improves the inspectors' effectiveness when compared to fault-based inspections. It was also reported that, educating developers on *errors* (as opposed to *faults*) made them less likely to make mistakes during the requirements development [6].

Motivation: While RET was found effective and useful, it was not meant to be complete and final product. A major drawback of RET was a lack of *human error* research on normal psychological processes that produce errors. A need for deeper investigation of human cognition failures was also highlighted by RET study results that reported *People Errors* (due to fallibilities of people involved) as a source of significantly large portion (i.e., up to 32%) of faults. These results motivated us to conduct an extensive psychological research on *human errors* in order to develop a human error taxonomy (HET). The resulting taxonomy (HET) is deeper (multi-level) and structured based on the theoretical human error types proposed by Reason's well-respected taxonomy of human errors [10]. Details of HET (Figure 1) appear in Section II.

Contribution: The current paper investigates whether HET can further enhance the inspection performance (as compared to RET) during an error based inspection. To answer this question, a control group experiment compared the fault detection *effectiveness* (# of faults), *efficiency* (faults/hour), and *usefulness* (subjects' feedback) of HET (Figure 1) vs. RET [2] during the inspection of SRS documents. In addition, the study provided insights into the nature of commonly occurring human errors at

different points during the requirement development process. More details appear in Sections III-V.

II. HUMAN ERROR TAXONOMY

Human error taxonomy (HET) was developed based on the systematic literature review (SLR), and in direct interaction with human error experts (via human error workshop – WAHESE 2015; <http://humanerrorinse.org/workshops>). To enable the contribution from Cognitive Science perspective of human errors, HET was created under the guidance of a Cognitive Psychologist (who is also a co-author on this paper). As a result, Reason’s [10] human error classification system (which has also been adapted in other domains) of *slips*, *lapses*, and *mistakes* was used to classify requirement phase human errors. When faced with a situation that requires problem solving, humans perform two cognitive activities: planning and execution. Reason [10] calls the cognitive failures (or human errors) that occur during planning, *mistakes*, and the cognitive failures that occur during execution, *slips* and *lapses*.

Reason’s slips are results of inattention while executing routine tasks. Slips are exemplified in common day to day activities like typing something incorrectly or “fat-fingering” due to not paying attention. Lapses occur when executing well-planned tasks, but are failures of memory. For example, having planned to repair a broken machine-part, but forgetting it due to interruption (e.g., taking a lunch break) is a common lapse. Mistakes are planning failures and occur when trying to solve an unfamiliar problem. An example of a mistake is a physician misdiagnosing a patient due to either not properly studying this patient’s symptoms or having no experience whatsoever with the symptom’s exhibited by this specific patient. In order to aid reader’s understanding, detailed information and examples of slips, lapses, and mistakes are provided in [7].

After we selected the right human error classification system (i.e. Reason’s slips, lapses, and mistake), the software engineering literature was surveyed to identify requirement phase human errors. While selecting requirements phase human errors, we strictly adhered to the following definition of human errors: *a human error is a flaw in human thought process*. For example, misunderstanding user’s need is an *error*, while

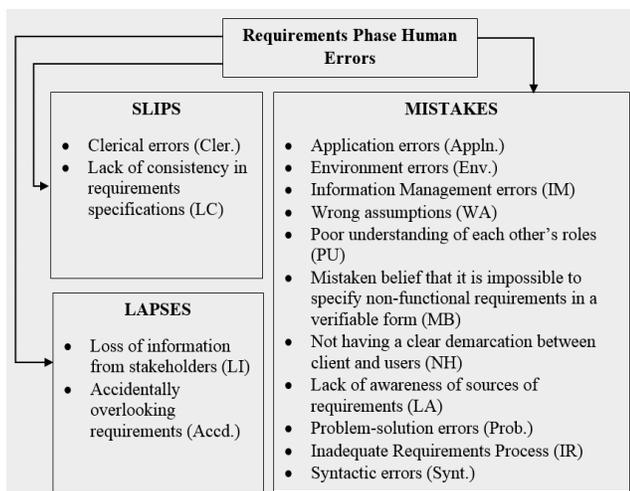


Figure 1. The Human Error Taxonomy (HET)

writing incorrect requirements specification due to that misunderstanding is a *fault*. A total of 15 requirements phase human errors identified from Software Engineering literature were classified as a slip, a lapse, or a mistake. The resulting taxonomy is shown in Figure 1 with details in [7].

III. EXPERIMENT DESIGN

The main goal of this experiment was to understand if changes made to the error taxonomy (i.e., development of HET) offer an improvement over RET (a proven verification technique) during the requirements inspection. To do so, a randomized pre-test post-test control group experiment was planned and executed in controlled settings. The control group used RET [2], whereas the experimental group used the newly developed HET to perform requirements inspection.

A. Experiment Methodology

This section describes research questions (RQ’s), study variables, and artifacts with complete package available at http://humanerrorinse.org/Studies/2015/Fall_NDSU_Experiment_1/index.htm.

1) RQ’s: The following four RQ’s were investigated.

RQ 1: Which error taxonomy (HET vs RET) provides better fault detection effectiveness (# of faults found) and efficiency (faults/time) during the requirements inspection?

RQ 2: Which error taxonomy (HET vs RET) is perceived more useful for finding faults during the inspection?

RQ 3: What insights into the major type of *human errors* can be used to further enhance the inspection performance?

RQ 4: Can an inspector’s performance on a pre-test accurately predict their inspection performance on a real project?

2) Variables:

Table I provides study variables and their descriptions.

TABLE I. STUDY VARIABLES

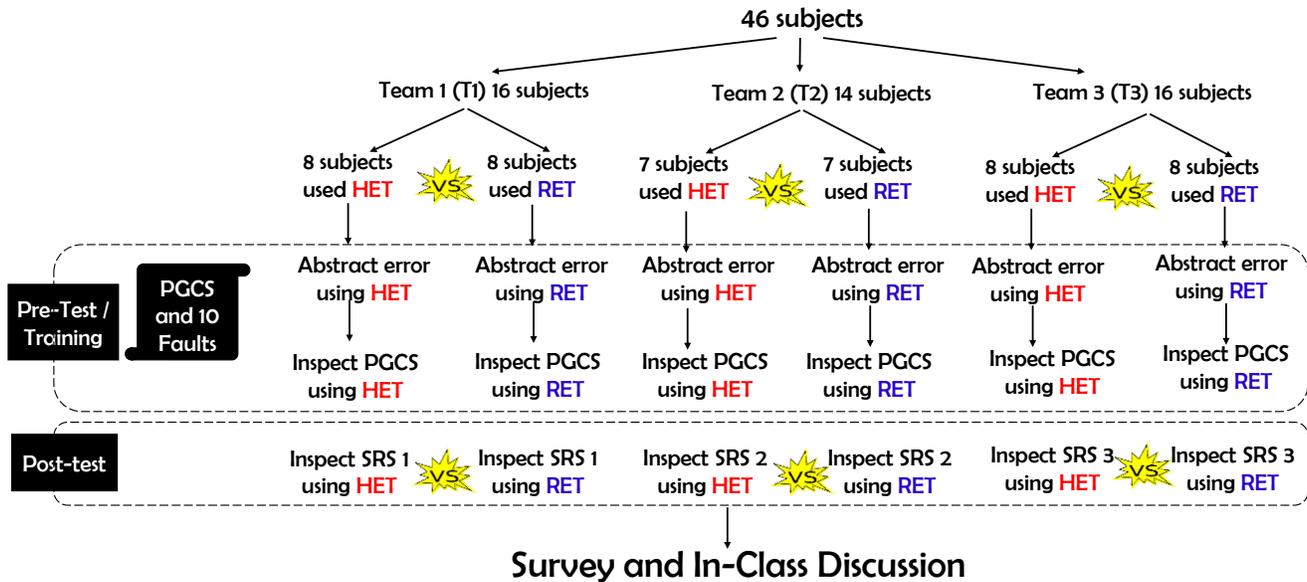
Independent Variables	Description
Pre-test	measures performance of subjects in a practice inspection exercise using HET/RET.
Effort Spent	time spent by the individual subjects to perform the inspection tasks.
Usefulness	perceived usefulness of HET/RET
Dependent Variables	Description
Effectiveness	Number of faults found by each subject
Efficiency	Number of faults found per hour

3) Subjects:

46 computer science students, enrolled in *Principles of Software Engineering* course at North Dakota State University (NDSU) participated in this study. The course required students to work in teams (teams were selected by the instructor prior to this study) to develop requirement artifacts for different software systems. To enable a comparison between HET vs. RET, researchers randomly divided subjects in each team into two equal groups (a *control group* that used RET and an *experiment group* that used HET). Figure 2 shows the division of subjects into three teams (e.g., team 1 had 16 subjects) and subdivision of each team into treatment groups (8 used RET and 8 HET).

4) Artifacts:

During the training (*pre-test*), participating subjects (23 in experimental group and 23 in control group) were trained on



their respective taxonomies (HET for experiment and RET for control group) by having them perform an error based inspection of an externally produced SRS document that was seeded with 30 realistic faults. The SRS used during the training described requirements for a Parking Garage Control System (PGCS).

During the *post-test*, subjects inspected the SRS's that they had developed (as part of a team) during the course of the semester. Table II provides a brief system description for the SRS's created by each team. As mentioned earlier and shown in Figure 2, half of the subjects within each team inspected their own SRS using HET (e.g., 8 in team 1) or RET (other 8 in team 1), depending on the treatment group they were assigned (and trained) during the pre-test.

TABLE II. TEAMS AND SYSTEM DESCRIPTIONS

Team#	System Name and Description
1	Fly-by: airline reservation and travel management platform
2	Campus Reconnection: college student information management and course management system.
3	FaceSpace: music streaming based on analysis of music played by user and other user-generated metrics.

B. Experiment Procedure

Figure 2 shows the comparison between the control group (RET) and experimental group (HET) at *pre* and *post-test*:

1) Pre-test Steps: Training on HET and RET

During the *Pre-test*, subjects were trained on HET and RET by having them perform an error-inspection on PGCS SRS document and report errors and faults. Details appear below:

Training: Subjects were trained on the importance of requirements inspections and different type of requirement faults. Next, in two separate sessions, 23 subjects were trained on HET and 23 subjects were trained on RET to teach them about the *error abstraction* using HET/RET (i.e., how to identify errors), and using the error information to perform *fault inspection* (i.e., how to identify new faults).

Error Abstraction - After the training, subjects were provided with 10 PGCS SRS faults (chosen randomly from 30 seeded faults). Subjects then used HET/RET to abstract and classify errors from 10 given faults. This step resulted in 46 error forms (23 for RET and 23 for HET).

Fault Inspection - Next, subjects used the abstracted error information (from error forms) to find additional faults in the PGCS SRS (i.e. subjects inspected PGCS SRS using errors). This step resulted in 46 individual fault-forms (23 for HET and 23 for RET) containing new faults in PGCS document.

2) SRS Development (listed in Table II)

Subjects then worked in their respective teams (three teams) to develop requirements for different systems.

3) Post-test Steps: Error-Inspection on self-created SRS

During the *post-test*, every subject inspected their own SRS (they had developed as a team) using the technique that were trained during the pre-test (HET or RET) and reported faults. For example, of 16 subjects in Team 1, 8 used HET while the other 8 used RET to inspect the “Fly-by” SRS. This step produced individual fault-forms from HET and RET for each team.

Post-study Survey and Focus Group: The experimental group and the control group subjects rated HET and RET across various usefulness categories on a 5-point scale (ranging from “1 – not useful” to “5 – very useful”). A focus group discussion was conducted with the goal of understanding the problems faced by subjects while using HET/RET to find faults.

IV. DATA ANALYSIS AND RESULTS

The results from analysis of data collected during the study run and organized around four RQ's listed in III.A.

A. Fault Detection Effectiveness of HET vs RET (RQ 1a):

We compared the fault detection effectiveness of the experimental group subjects (who used HET) vs. control group subjects (who used RET), during the inspection of their self-created SRS documents. The fault reporting forms during the post-test were analyzed separately for each team (after removing

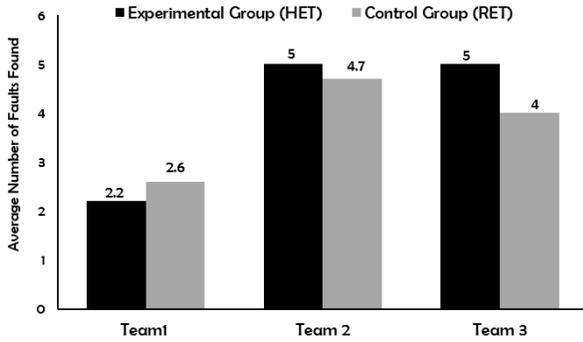


Figure 3. Comparison of average number of faults

the fault positives) to enable this comparison. Figure 3 reports the resulting comparison of the average number of faults detected by the experimental group subjects' vs the average number of faults detected by the control group subjects within each team. For example, for Team 3, experimental group subjects found an average of five (5) faults and the control group subjects found an average of four (4) faults.

For teams 2 and 3, as shown in Figure 3, the experimental group subjects generally found more faults than the control group subjects, however these results were not significant (based on an independent samples t-test). The inconsistency with Team 1's results was analyzed and we found that many subjects (6 out of 16) from Team 1 did not participate in this experimental task (didn't inspect using their assigned technique). Therefore, low fault count for Team 1 (irrespective of HET/RET) was found to be due to lack of participation and understanding of the tasks involved.

The results show that, when properly motivated, HET can provide better fault detection effectiveness as compared to RET.

B. Fault detection efficiency of HET vs RET (RQ 1b):

We also compared the *efficiency* (faults per hour) of the experimental and control group subjects within each team. The timing data (start and end times, the time each fault was fault, breaks) reported by subjects in their fault forms was used to calculate the fault rate. The *average fault rate* comparison of treatment groups is shown in Figure 4.

As shown in Figure 4, subjects using HET (experiment group), found faults at much faster rate when compared to the subjects who used RET (control group). The results from an independent samples t-test showed that although subjects using HET were consistently more efficient for all three teams, the improvement was not significant.

This was the very first investigation of HET whereas RET has been validated and improved through a series of comprehensive empirical analysis [3-5]. That said, subjects using the HET not only found more faults, they found it at a much faster rate. This also shows that, more effort (time spent) is needed to use RET to match the productivity that can be achieved with HET. That is, the learning curve for HET is smaller than that of RET, which justifies the motivation for the development of HET.

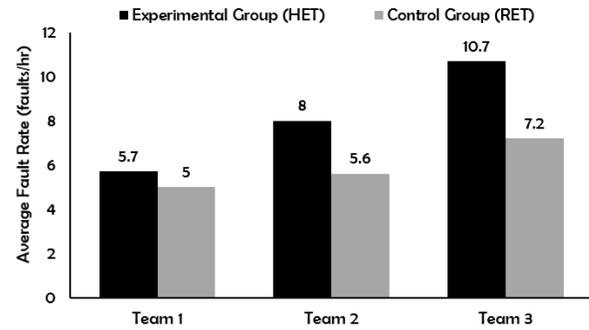


Figure 4. Comparison of average fault rate or efficiency (faults/hour)

C. Perceived Usefulness of the Two Error Taxonomies (RQ 2)

Next, we evaluated subjects' self-reported data (collected during post-study survey) regarding the *usefulness* of HET and RET on five essential attributes: (1) *usability-usab.*, (2) *orthogonality (Orth.* - a lack of overlap amongst error classes, (3) *usefulness* of error taxonomy in locating faults - *Usef.*, (4) *confidence* that taxonomy represented real RE problems - *Conf.*, and (5) *worthiness* of the effort spent in using the taxonomy - *Worth.* A 5-point Likert scale ranging from 1-*Strongly Disagree* to 5- *Strongly Agree* was used to evaluate these attributes of both taxonomies. This comparison was performed for 23 subjects who used (and rated) HET vs. 23 subjects who used RET.

TABLE III. HET VS. RET COMPARISON USING 5-POINT SCALE

	Usab.	Orth.	Usef.	Conf.	Worth.
HET	2.94	2.94	3.89	3.89	3.33
RET	2.68	3.0	3.42	3.79	3.21

Table III compares of average ratings of HET and RET, across five attributes. Overall, HET was rated more positively (greater than 3 – midpoint on a 5-point scale) than RET in terms of *usefulness*, *confidence* and *worthiness* aspects of error based inspections. The shaded cells in Table III for HET, were rated significantly greater than the midpoint of scale ($p < 0.001$) based on the results from one-sample t-test. However, RET received better feedback rating for the attribute, *orthogonality of error classes*. This was expected because unlike RET, HET includes errors within an error type (e.g., *Application error* – an error class under *Mistake* in Figure 1) that can happen at different points during the requirement development (i.e., elicitation, analysis, and verification). We plan to explain this in more detail during the training (in future studies) and also plan to provide more examples of errors (in HET training document) to make it more easy to use in future.

D. Insights into Human Errors during Requirements Development (RQ 3).

While human error research seems worthwhile (based on these results), we wanted to gain useful insights into the major sources of requirement faults. This in turn can help researchers (and practitioners) to focus on more frequently occurring human errors and reduce their frequency through interventions (e.g., a checklist or a tool-based assist). To perform this analysis, faults

and errors reported by the subjects (using HET) during the post-test (inspection of self-created SRS's) were used.

Figure 5 reports the results of this analysis in terms of the percentage contribution of *Slips*, *Lapses* and *Mistakes* to the overall faults reported by each team while inspecting their SRS. Results from Chi-square test showed that, for all three teams the observed contribution of error types (slips, lapses, and mistakes) were significantly different ($p < 0.001$) from uniform distribution (33.33%). We also analyzed the contribution of each of 15 error classes within HET (Figure 1) to evaluate the most common types of *slips* and *mistakes* that the developers can be made aware prior to the development. The resulting contributions of each of 15 error classes (towards total reported faults) is shown in Table IV (highlighted fields reported higher contributions).

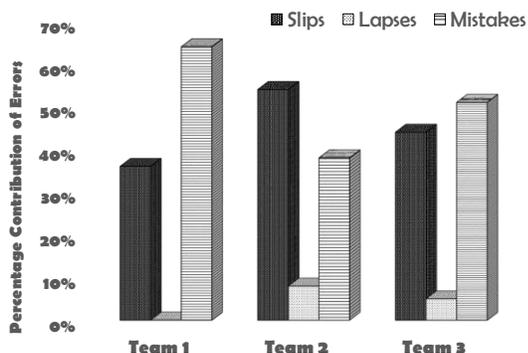


Figure 5. Percentage Contribution of Slips, Lapses and Mistakes

Major insights gained from this analysis is follows:

- Based on these results, *Slips* and *Mistakes* (as opposed to *Lapses*) are a major source of requirement faults.
- *Clerical Errors* (a sub class of *Slips*) and *Application errors* (a sub class of *Mistakes*) occurred at a higher frequency as compared to other error classes.
- *Clerical errors* occurred (reported retrospectively by the subjects) during the elicitation phase. *Application errors*, on the other hand occurred at different points, due to the misunderstanding of particular aspects of problem solution, and led to omission of relevant information from requirements document.

These results are different from the human error results reported in Cognitive Psychology (CP). CP literature reports *Slips* and *Lapses* contributing up to 60% of total errors [15], whereas the results from this study showed that *Slips* and *Mistakes* contributed around 90% of total faults (Team 1, did not report any lapses). A possibility could be that, since subjects are evaluating their own work, they might be less likely to report

lapses (i.e., memory related failures). We plan to conduct multiple studies to be able to generalize the findings.

E. Prediction of Performance using Pre-test Data (RQ 4).

The performance of subjects during the *pre-test* (# of faults found during the PGCS inspection using HET/RET) was correlated with their performance during the post-test (i.e., # of faults found in their own SRS using HET/RET). The goal was to analyze whether the performance of subjects using HET (or RET) while inspecting their own SRS (i.e., post-test) could be predicted by their performance of respective taxonomies during the training. To perform this analysis, the number of actual faults found by subjects in the experiment (HET) and control (RET) group were compared at pre (PGCS SRS) and posttest (self-created SRS). Figure 6 shows the correlation between pre and posttest performance for the experimental and control groups.

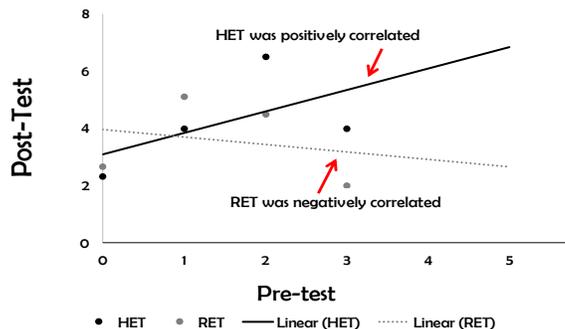


Figure 6. Pre-test vs post-test performance

Based on these result, the experimental group (HET), displayed a positive correlation between pre-and post-test performance whereas control group (RET) subjects showed a negative correlation between pre and post-test performance. This means that when using HET (rather than RET), project managers can help predict the inspectors' performance during the real project. This is objective information for project managers to help plan the inspection process.

V. VALIDITY THREATS

The authors tried to address some of the validity threats. The *selection* threat was reduced by randomizing the experimental group and control group subject placement. This also ensured that treatment groups were equivalent. The threat to *external* validity when using a toy requirements document was also removed by the fact that students inspected real requirements specification documents that they developed and contained naturally occurring faults. However, a threat remains that there might be additional faults (that were not reported by subjects) present in SRSs and we plan to perform additional analysis on

TABLE IV. CONTRIBUTION OF ERROR CLASSES TO TOTAL HUMAN ERRORS WHICH LED TO FAULTS

	Slips		Lapses		-----Mistakes-----										
	Clerical	LC	LI	Accd.	Appl.	Env.	IM	WA	PU	MB	NH	LA	Prob.	IR	Synt.
Team 1	36%				9%			27%					9%	18%	
Team 2	54%			8%	19%		4%		4%				12%		
Team 3	41%	3%		5%	10%	8%	3%	8%	5%					8%	10%

these SRSs. Also, the study was conducted in a classroom setting that is not representative of time and pressure in real settings. We plan to address this threat in future studies.

VI. DISCUSSION OF RESULTS

In this section, each original research question is revisited to discuss the implications of the results of data analysis.

RQ1: In terms of *effectiveness*, the subjects using HET found more faults compared to the subjects who used RET for two out of three teams. In terms of *efficiency*, subjects using HET across all three teams found faults at a much faster rate when compared to the subjects using RET. A major goal of this study was to evaluate if students' learning curve on identifying human errors and corresponding faults can be enhanced (from RET), and the *efficiency* results justifies the construction of HET. From project managers' perspective, an *efficient* inspection approach allows them to improve software quality without increasing overall projects costs.

RQ2: The results showed that while both error taxonomies were rated favorably, HET received slightly better feedback in four out of the five categories. Subjective feedback given by the experimental group subjects also shed light on the fact that NH and LA error classes under the *mistakes* (NH - *not having a clear demarcation between clients and users*; and LA - *lack of awareness of sources of requirements*) were hard to distinguish and need to be either merged together or supplied with more examples. Based on their subjects' feedback, the process of using error information to find faults can be formalized to further improve the usefulness of HET.

RQ3: The results showed that Slips and Mistakes occurred at a higher frequency as compared to Lapses. Further, *Clerical* errors and *Application* errors were most commonly occurring slips and mistakes respectively. Additional investigations are needed to evaluate if this lack of occurrence of *lapses* can be generalized to all software systems.

RQ4: The results showed that when using HET, an inspector's performance during the training can help predict their performance during an inspection on live project. This can help project managers to staff inspectors and better plan inspection process at their organizations.

VII. CONCLUSION AND FUTURE WORK

This study reported the newly developed Human Error Taxonomy (HET) and compared it against the seasoned Requirement Error Taxonomy (RET), with regards to the fault detection effectiveness and efficiency of requirement inspectors when using HET vs RET. The overall results show that HET not only helped inspectors find more faults, but also at a faster rate. This encourages further research into the usage of human error abstraction and classification (using HET) for requirements defect detection. Subjects, in their feedback reported the need for a more systematic process of using errors to find faults. The authors plan to do more studies to collect extensive data set on human errors and its impact on the requirement. We also intend to develop a tool that will assist the inspectors during the error abstraction process. The tool will be developed to help inspectors abstract human errors from the perspective of -

"which requirement activity (elicitation, analysis, specification, verification, or management) did the human error occur in?"

The authors expect a significant improvement in HET-based inspection approach with the inclusion of such aforementioned tools.

ACKNOWLEDGMENT

This study was supported by National Science Foundation Awards 1423279 and 1421006. The authors would like to thank the participating students enrolled at NDSU and the course instructor for their help during the study run.

REFERENCES

- [1] F. Lanubile, F. Shull, and V. R. Basili, "Experimenting with error abstraction in requirements documents," in 5th International Symposium on Software Metrics, Bethesda, 1998.
- [2] G. S. Walia and J. C. Carver, "A systematic literature review to identify and classify software requirement errors," *Inf. Softw. Technol.*, vol. 51, no. 7, pp. 1087–1109, 2009.
- [3] G. S. Walia, J. C. Carver, and P. Thomas, "Requirement Error Abstraction and Classification: An Empirical Study," in 5th International Symposium on Empirical Software Engineering, New York, 2006, pp. 336–345.
- [4] G. S. Walia, J. C. Carver, and T. Philip, "Requirement error abstraction and classification: A control group replicated study," in Proceedings - International Symposium on Software Reliability Engineering, ISSRE, Sweden, 2007, pp. 71–80.
- [5] G. S. Walia and J. C. Carver, "Evaluating the use of requirement error abstraction and classification method for preventing errors during artifact creation: A feasibility study," in Proceedings of International Symposium on Software Reliability Engineering, ISSRE, San Jose, 2010, pp. 81–90.
- [6] G. S. Walia and J. C. Carver, "Using error abstraction and classification to improve requirement quality: Conclusions from a family of four empirical studies," *Empir. Softw. Eng.*, vol. 18, no. 4, pp. 625–658, 2013.
- [7] V. K. Anu, G. S. Walia, W. Hu, J. C. Carver, and G. Bradshaw, "Usefulness of Human Error Taxonomy as an Effective Requirements Inspection Technique: An Empirical Investigation," TECHNICAL REPORT, Fargo, ND, 2015, http://humanerrorinse.org/Studies/2015/Fall_NDSU_Experiment_1/index.htm
- [8] R. Chillarege, I. S. Bhandari, J. K. Chaar, M. J. Halliday, B. K. Ray, and D. S. Moebus, "Orthogonal Defect Classification: A Concept for In-Process Measurements," *IEEE Trans. Softw. Eng.*, vol. 18, no. 11, pp. 943–956, 1992.
- [9] M. Leszak, D. E. Perry, and D. Stoll, "A case study in root cause defect analysis," in Proceedings of the 22nd international conference on Software engineering - ICSE, Limerick, Ireland, 2000, pp. 428–437.
- [10] J. Reason, *Human error*. Cambridge, U.K.: Cambridge University Press, 1990.
- [11] S. A. Shappell and D. A. Wiegmann, "Applying Reason: the human factors analysis and classification system (HFACS)," *Hum. Factors Aersp. Saf.*, vol. 1, no. 1, pp. 59–86, 2001.
- [12] D. Wiegmann and C. Detwiler, "Human Error and General Aviation Accidents: A Comprehensive, Fine-Grained Analysis Using HFACS," *Security*, pp. 1–5, 2005.
- [13] B. Boehm and V. R. Basili, "Software Defect Reduction Top 10," *Computer*, vol. 34, no. 1, pp. 135–137, 2001.
- [14] O. Laitenberger, "A Survey on Software Inspection Technologies," in *Handbook on Software Engineering and Knowledge Engineering*, vol. 2, 2002, pp. 517–555.
- [15] A. Esgate, D. Groome, and K. Baker, *An Introduction to Applied Cognitive Psychology*. Psychology Press, 2005.

Requirements Engineering Related Usability Techniques Adopted in Agile Development Processes

Daniel A. Magües

Escuela Politécnica Superior
Universidad Autónoma de Madrid
Madrid, Spain
daniel.magues@estudiante.uam.es

John W. Castro

Departamento de Ingeniería Informática
y Ciencias de la Computación
Universidad de Atacama
Copiapó, Chile
john.castro@uda.cl

Silvia T. Acuña

Escuela Politécnica Superior
Universidad Autónoma de Madrid
Madrid, Spain
silvia.acunna@uam.es

Abstract—Context: Over the last decade there has been a growing interest in the integration of agile software development process (ASDP) and user-centred design (UCD). However, there are no papers that study which usability techniques related to requirements engineering are being adopted in the ASDP, and there are no formalized proposals for their adoption. **Objective:** Identify which techniques related to requirements engineering activities are being adopted in the ASDP and determine how they are being adopted. **Method:** We have conducted a systematic mapping study (SMS) to retrieve the literature reporting the application of usability techniques in the ASDP. We analysed these techniques using a catalogue of techniques compiled by software engineering researchers. We then determined the manner in which the techniques that are being used in the ASDP were adopted. **Results:** The agile community is very much interested in adopting usability techniques. The most used techniques are Personas, contextual inquiry and prototyping. **Conclusions:** This research offers an overview of the adoption of usability techniques related to requirements engineering in ASDPs and reports how they are being adopted. We found that some of the techniques are being adapted for adoption.

Keywords- agile software development; user-centred design; systematic mapping study; usability; usability techniques

I. INTRODUCTION

The integration of agile software development processes (ASDP) (for example, XP [1] and SCRUM [2]) and the user-centred design process has been a focus of research over the last few years [3][4][5]. This growing interest is explained by the fact that UCD is useful for understanding the needs of future system users and how the software can provide support for their goals and activities in order to improve usability and user satisfaction with system interaction. However, such features are not usually taken into account during requirements elicitation in the ASDP [3][5]. Usability is a quality attribute of software system use and relies not only on the appearance of the user interface but also on how the user interacts with the system [6].

The ASDP recommends that, instead of collecting all the requirements at the start of the project, they should be elicited during just-in-time cycles for each increment as the software is built [2]. According to some authors, this is a good strategy for handling and prioritizing “emerging requirements” (together with stakeholders) and adapting team workload accordingly

[2][7]. On one hand, in the particular case of SCRUM, the requirements are accommodated in a product backlog as user stories. These user stories are living entities because they are constantly changing [2]. On the other hand, UCD places the user at the centre of requirements analysis, design and evaluation activities in order to improve the usability of the final product [3]. UCD professionals apply UCD techniques and methods as part of a collaborative and iterative process [8][9]. Therefore, UCD professionals and teams enacting the ASDP are potentially well-matched, and their alignment could improve user experiences.

However, the usability requirements elicitation principles and practices applied in agile processes have been found to be wanting [4]. This occurs when, for example, the usability requirements are elicited in situ from customers that have a perfect understanding of the needs of the system but not of the different types of end users. This is an obstacle to the identification of potential usability problems facing novice end users [10][11]. One of the principles of UCD, on the other hand, is to understand all user profiles [11]. Many authors working in the human-computer interaction (HCI) domain claim that UCD professionals have to adapt their mind-set to the agile process [2][12][13][14]. But this is by no means straightforward for two reasons. First, many UCD professionals work as part of teams that are separate from and provide support to the agile development team. Therefore, they have a different culture. Second, many professionals have had to develop their own strategies to align UCD practices with the ASDPs adopted by their organization [15]. Thus, the adaptation of usability techniques and methods is based on the experience of UCD professionals, and many such usability techniques require time and resources that an agile process cannot afford. Additionally, agile processes do not provide any guidelines for such adaptations.

Therefore, there is a need for research into which usability techniques can be adopted in ASDP requirements engineering activities. Some researchers have completed empirical studies [16][17][18], whereas a few have conducted comprehensive literature reviews [4][19] in order to identify the usability techniques that are being adopted in the ASDP. The goal of our research is to ascertain the current state of usability in the ASDP from a broader perspective. In order to analyse the benefits of usability in the ASDP and identify which usability techniques related to the requirements engineering activity are

being adopted and how, we have conducted a systematic mapping study (SMS) of usability and ASDPs.

The literature on the integration of usability techniques into the ASDP is composed of a set of assorted papers that study different issues related to the topic. We identified two problems from these papers. First, there are not many papers that study the topic as a whole and report the current state of integration [11][19][20]. Second, there are no formalized proposals for adopting usability techniques in the ASDP [18][21][22] that establish guidelines for each adopted technique. Therefore, the research problem addressed in this paper is review the literature in order to identify which usability techniques related to requirements engineering activities are being adopted in the ASDP.

This paper is organized as follows. Section 2 discusses the catalogue of HCI techniques that we used as a baseline for investigating the usability techniques that the agile community is using. Section 3 describes the research method. Section 4 reports the usage of usability techniques related to requirements engineering in ASDPs. Section 5 discusses how the usability techniques have been adopted by the ASDPs. Finally, section 6 reports our conclusions and future work.

II. BACKGROUND

In order to find out which techniques are being used in agile development projects to produce usable systems, we first need to identify the universe of HCI techniques. This is far from straightforward. There are a wide range of HCI techniques, where the same technique may be referred to differently depending on the author and there may be more than one variant on the same technique. Fortunately, other software engineering researchers have already taken the trouble to compile a catalogue of HCI techniques [24]. Ferré [24] compiled a list of techniques from the recognized HCI sources. There follows a very brief summary of this catalogue which should help readers to follow the remainder of the paper analysing which usability techniques are used in ASDP and how they are being adopted.

According to Ferré [24], the most representative activities of the HCI process are use context specification, usability specifications, product concept development, prototyping, interaction design, and usability evaluation. Ferré [24] maps these activities (and their respective associated techniques) to SE development stages. HCI activities are in some cases integrated into existing SE activities. For example, the usability specifications activity can be integrated into requirements specification. In other cases, however, further activities that are not usually carried out in a non-user-centred development process, such as interaction design, have to be added. These extra activities will be referred to as usual in the HCI community. HCI activities have been mapped taking into account SE development stages: requirements engineering, design and evaluation. HCI techniques in the catalogue are classified according to the meaning of requirements engineering, design and evaluation for SE. For our research, we only had to consider techniques related to the requirements engineering activity. The HCI activities mapped to the requirements engineering activity are: use context

specification, usability specifications, product concept development and prototyping.

III. RESEARCH METHOD

We have conducted a SMS [23] to investigate the current state of the integration of agile processes and usability. The electronic databases (DB) used in the SMS were Scopus, ACM Digital Library and IEEE Xplore. The review covers papers published up until 15 October 2015. Fig. 1 illustrates the search string used in the SMS. We used different synonyms to extend the scope of the search. The inclusion criteria of the literature review state that the research papers should mention an issue related to usability and agile development. The search was divided into two stages. During the first stage, we examined the title, keywords and abstract to screen the 409 papers retrieved from all three DBs, of which we selected 172 as possibly relevant papers. Only papers that were written in English and whose abstract or title mentioned an issue related to the integration of agile processes and usability, a topic related to usability engineering or HCI techniques, or a question related to the UCD process were selected. Papers were rejected if they made no mention of anything to do with the integration of agile processes and usability or the UCD process. During the second stage, we read the abstract, introduction and conclusions to determine whether the paper described any type of integration between agile processes and usability through HCI techniques or practices. Finally, as a result of the second stage, we retrieved a total of 31 relevant papers (primary studies).

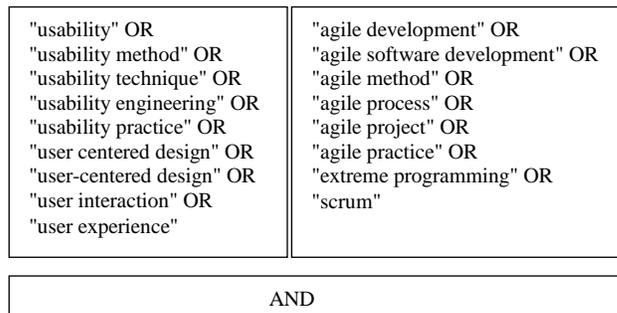


Figure 1. Keywords used for the search string.

The terms used in the SMS search string are the most commonly used by authors specializing in the field. However, other terms identifying other relevant papers may have been overlooked. Additionally, the papers were analysed and evaluated based on our opinions and experience. This means that other authors may have arrived at different conclusions about the same results.

All three members of the research team helped to define the search strategy. Two researchers designed and ran the searches and extracted the data. Later, the findings were discussed at meetings attended by all team members.

After analysing all the primary studies, we found that practice integration accounts for the adoption of specified HCI techniques. The papers classified as practice integration identify the adoption of usability techniques in the ASDP and vice versa.

The primary studies that we study in order to analyse the usability techniques adopted in the ASDP refers to practice integration (31 publications). Our reason for selecting this type of integration was that, because no profound process changes are called for, it would, we believe, be more practicable to integrate usability techniques into ASDPs, since this would require neither a major effort to train the teams enacting the ASDP nor additional investment in order to bring new roles into the team. After discussing each paper at meetings attended by all the research team members, we determined which usability techniques have been adopted in the ASDP, outputting the description given by the author. These techniques were analysed by comparing the manner in which the technique was applied in the ASDP with the original recommendation established by the HCI community. It took several rounds to identify these techniques. The process was complicated by the fact that the authors of the primary studies did not use the customary technique names. Therefore, we had to read the paper more thoroughly in order to identify technique to which the authors were referring. Through the description of the techniques adopted in the ASDP, the use of a catalogue of HCI techniques [24][25][26][27] (detailed in Section 4) and consultation with experts, we identified which usability techniques reported in the literature have been adopted in ASDP development projects. An expert in usability techniques participated in this identification process. This process yielded a preliminary classification of the different usability techniques related to requirements engineering adopted in ASDPs.

We then analysed how the usability techniques were adopted in ASDP requirements engineering activities. As a result of this analysis, we found that some techniques were adopted as per HCI recommendations and others needed adaptation. For each of the techniques adopted by the ASDP, we identified the adaptations made (some are discussed in Section 4). This turned out to be one of the most complex processes because of the sheer number of techniques and the fact that the primary studies did not compare the technique that they described with HCI recommendations. This comparison was necessary in order to be able to identify the adaptations made to the techniques.

IV. USE OF HCI TECHNIQUES RELATED TO ASDP REQUIREMENTS ENGINEERING

We have found that the agile community has adopted a number of usability techniques in development projects. We have classified the adopted usability techniques into two groups. The first group includes all the techniques that have been adopted as is, that is, have been applied as recommended by HCI. The second group includes the techniques that have had to be adapted for adoption. In order to identify the techniques adopted in ASDPs, we examined the papers in the practice integration group. We read each paper carefully to identify the names of all the techniques reported by the author and their respective description. Note that some authors report the adoption of more than one usability technique. After reading their respective descriptions, we then classified each technique adopted in ASDPs according to the HCI technique catalogue [24]. This task was carried out in conjunction with experts in the area as part working sessions aimed at identifying which catalogue techniques had been adopted by

ASDPs. Only the description given by the author of how the technique was adopted was considered for identification purposes. The name given to the technique was omitted because of the possibility of it distorting our classification. Generally speaking, the authors are not expert technique catalogue users. Table I summarizes the number of HCI techniques related to the requirements activity adopted in ASDPs. This summary includes the total number of existing techniques (according to the catalogue used [24]) and the percentage of such techniques that we identified as having been adopted by the agile community. Note that some of the techniques and their respective description required a more thorough analysis in order to identify the HCI technique to which they mapped in the catalogue.

TABLE I. PERCENTAGE OF HCI TECHNIQUES ADOPTED IN ASDPs

SE Development Stage		No. of HCI Techniques (following [24])	No. of Techniques Adopted by ASDPs
Requirements Engineering	Requirements Elicitation and Analysis	25	13 (52.00%)
	Requirements Specification	1	0 (0.00%)
	Requirements Validation	7	3 (42.86%)

Additionally, the HCI technique catalogue had to be expanded because ASDPs have adopted techniques listed in the catalogue, which, however, they have applied in SE development stages other than those specified in the HCI catalogue. For example, Losada et al. [28][29] propose designing questionnaires, surveys and interviews as part of the requirements elicitation activities.

We found that ASDPs have adopted 52% of the usability techniques related to requirements engineering. Of this group, Personas is the most used technique, followed by contextual inquiry and prototyping. Table II summarizes the HCI techniques related to requirements engineering activities adopted by ASDPs. For each technique, we specify the SE development stage to which it is related, a generic technique name, the name given by different authors in the HCI literature (the variants of the generic technique in italics), the name used by the ASDP authors, the references and the level of adoption (as is or with modifications). Note that, for reasons of space, Table II does not list the usability techniques related to requirements engineering that have not been adopted by the agile community in its development projects.

V. RESULTS AND DISCUSSION

In this section, we discuss how ASDPs have adopted usability techniques. Note that all mentions of usability techniques adopted in ASDP development projects throughout this research paper refer to techniques that have been adopted and then reported in the literature. We have classified each and every one of the usability techniques that have been adopted by ASDP developments. They have been classified according to a

HCI technique catalogue [24], analysing how the technique was applied. We have found that techniques are being adopted in two ways: usability techniques applied as is (i.e., the technique has been applied as recommended by HCI) and techniques that have been adapted (i.e., the technique has been somehow modified for application within agile developments).

The usability techniques adopted in ASDP requirements engineering activities are: contextual inquiry, ethnographical observation, card sorting, Personas, questionnaires, surveys and

interviews, essential use cases, task scenarios, task sorting, scenarios and storyboards, prototyping, inspections, cognitive walkthroughs, and evaluation by experts. Contextual inquiry, Personas and prototyping can be singled out as the most commonplace techniques. We have found that some techniques (e.g., contextual inquiry) have been adopted in agile developments both as is and with modifications. The type of adoption (as is, with modifications) depends on the particular features and resources of each project.

TABLE II. HCI TECHNIQUES RELATED TO SE REQUIREMENTS ENGINEERING ACTIVITIES ADOPTED IN ASDPS

SE Development Stage	Generic Technique Name	Technique Name Given by HCI Authors	Name Used by ASDP Authors	Ref.	Application Type	
Requirements Elicitation and Analysis	Contextual Inquiry	Contextual Inquiry	Contextual analysis	[32]	As Is	
			Context of use	[34]	As Is	
			Contexts of use	[31]	As Is	
		Contextual Interviews	Contextual inquiry	[30]	With modifications	
			Context inquiry	[16]	As Is	
			Contextual inquiry	[35]	As Is	
	Contextual Inquiry		[36]	As Is		
	Ethnographical Observation	Ethnography	Ethnographic research	[16]	As Is	
			Observation	[28]	As Is	
		Ethnographical Observation	Observation	[29]	As Is	
	Card Sorting	Card Sorting	Card sorting	[10]	With modifications	
			Card sorting	[25]	With modifications	
	User Analysis	Personas	Personas	Personas	[16]	As Is
				Personas	[27]	With modifications
				Personas	[37]	With modifications
				Personas	[31]	As Is
				Lightweight personas	[38]	With modifications
				Personas	[33]	With modifications
				Persona	[39]	As Is
				Persona	[36]	As Is
		Extreme personas	[40]	With modifications		
		Questionnaires, Surveys and Interviews	Questionnaires, Surveys and Interviews	Questionnaires	[29]	As Is
				Interviews	[29]	As Is
				Questionnaires	[28]	As Is
				Interviews	[28]	As Is
		Task Analysis	Essential Use Cases	Essential Use Cases	Essential use cases	[34]
	Develop Product Concept	Task Scenarios	Task Scenarios	Usability user stories	[30]	With modifications
				Usability user stories	[31]	As Is
				Usability user stories	[43]	With modifications
		Task Sorting	Task Sorting	Task models	[30]	With modifications
		Scenarios and Storyboards	Scenarios	Scenario based descriptions	[34]	As Is
				Scenarios	[10]	With modifications
Scenario based approaches				[38]	As Is	
User scenarios	[33]			With modifications		
Prototyping	Prototyping	Prototyping	Prototypes	[29]	As Is	
			Prototypes	[28]	As Is	
			Prototypes	[26]	As Is	
		<i>Paper Prototypes</i>	Paper prototypes	[31]	As Is	
		<i>Scripted Prototypes</i>	Mock-ups and prototypes	[34]	As Is	
		<i>Wizard of Oz Prototypes</i>	Wizard of Oz testing	[41]	With modifications	
Requirements Validation	Inspections	<i>Collaborative Inspections</i>	Usability inspection evaluations	[42]	With modifications	
	Cognitive Walkthrough	Cognitive Walkthrough	Informal cognitive walkthrough	[17]	With modifications	
	Evaluation by Experts	Evaluation by Experts	Peer review	[42]	With modifications	

As mentioned above, we found that some usability techniques are being adopted thanks to adaptations. For example, the HCI discipline stipulates that, in order to apply contextual inquiry, all the people involved in applying the technique (developers and users) must have previous training and the team must participate in the multidisciplinary meeting from beginning to end. However, agile team members generally do not have previous knowledge of applying usability techniques and developers have limited time and work to a tight schedule. We have found that, in order to solve these two problems, Beyer et al. [30] adapt contextual inquiry in two ways. First, the user interface team compensates for the training gaps of the other participants. Second, due to agile development time constraints, developers are not involved until the end of the multidisciplinary meeting so as not to affect team work in the preceding sprint.

The contextual inquiry technique has also been adopted as is. This technique has been used at the start of agile projects because it is capable of identifying users, their backgrounds, their motivations, their responsibilities and their roles [31]. Besides, it also yields enough information to generate the best possible upfront design and thus conforms to ASDP just-in-time analysis practices [32].

The most often used technique, Personas, has been adopted with modifications. We have found that there is a tendency to apply the technique iteratively as proposed by ASDPs developing the model as necessary for the functionality under development [33]. It has also been applied as is and used as a means of communication with developers to identify the different user types and understand their needs and perspectives [31].

The prototyping technique has been adopted as is in most cases because it is considered to be a lightweight technique for use in ASDPs. Low-fidelity paper prototypes, which do not require many resources or much time [28], are built in the early and intermediate stages and may be submitted to users for evaluation [34].

VI. CONCLUSIONS

In this research we determined which usability techniques are being adopted in agile development projects and looked at how they are being adopted. Additionally, we identified how some ASDPs are adapting the techniques for adoption. The most commonly used techniques are Personas, contextual inquiry and prototyping. On one hand, many HCI techniques related to requirements elicitation and analysis are naturally lightweight (e.g. paper prototypes), whereas others require adaptations to be able to be adopted in some agile projects (e.g. Personas). However, these are neither widespread nor prescriptive adaptations. On the other hand, as the process involves users, the activities need to be conducted in a methodical and structured manner. This can be achieved by adopting usability techniques, but ASDPs do not provide any guidelines for adoption.

There are three main ASDP adaptations of HCI techniques. The first is to conduct user testing outside laboratories. The second is to divide persona creation into several iterative cycles. The third is to abridge heuristic checklists and substitute

agile team roles such as product owner for experts in the heuristic evaluation technique.

Our results suggest that the agile development community is starting to adopt usability techniques in its development projects. However, a general, prescriptive and systematic proposal enabling agile development teams to adopt usability techniques in their development projects is missing in the literature. On this ground, there is a need for further research into this issue.

As future research, we intend to extend the paper searches to cover other databases, like, for example, SpringerLink and ScienceDirect. The aim is to increase the number of relevant papers retrieved, as we have found that there is a lot of interest in this topic within the scientific community.

ACKNOWLEDGMENT

This research was funded by the Spanish Ministry of Education, Culture and Sports FLEXOR and "Realizando Experimentos en la Industria del Software: Comprensión del Paso de Laboratorio a la Realidad" projects (TIN2014-52129-R and TIN2014-60490-P, respectively) and the eMadrid-CM "Investigación y Desarrollo de Tecnologías Educativas en la Comunidad de Madrid" project (S2013/ICE-2715).

REFERENCES

- [1] K. Beck, "Embracing change with extreme programming", *Computer*, vol. 32, n° 10, 1999, pp. 70-77.
- [2] M. Cohn, *Succeeding with Agile: Software Development Using Scrum*, Pearson Education, 2009.
- [3] S. Humayoun, Y. Dubinsky and T. Catarci, "A three-fold integration framework to incorporate user-centered design into agile software development", *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 6776 LNCS, 2011, pp. 55-64.
- [4] D. Salah, R. Paige and P. Cairns, "A systematic literature review for Agile development processes and user centred design integration", in *proceedings of the ACM International Conference*, 2014, pp. 5-14.
- [5] A. Wale-Kolade, P. Nielsen and T. Päiväranta, "Integrating usability practices into agile development: A case study", in *proceedings of the 23rd International Conference on Information Systems Development*, 2014, pp. 337-347.
- [6] N. Juristo, A. Moreno and M. Sanchez-Segura, "Guidelines for Eliciting Usability Functionalities", *IEEE Transactions on Software Engineering*, vol. 33, n° 11, 2007, pp. 744-758.
- [7] L. Cao and B. Ramesh, "Agile requirements engineering practices: An empirical study", *IEEE Software*, vol. 25, n° 1, 2008, pp. 60-67.
- [8] S. Chamberlain, H. Sharp and N. Maiden, "Towards a framework for integrating agile development and user-centred design", *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 4044 LNCS, 2006, pp. 143-153.
- [9] D. Fox, J. Sillito and F. Maurer, "Agile methods and user-centered design: How these two methodologies are being successfully integrated in industry", in *proceedings of the Agile Conference*, 2008, pp. 63-72.
- [10] D. Kane, "Finding a place for discount usability engineering in agile development: throwing down the gauntlet", in *proceedings of the Agile Development Conference*, 2003, pp. 40-46.
- [11] O. Sohaib and K. Khan, "Integrating usability engineering and agile software development: A literature review", in *proceedings of the International Conference on Computer Design and Applications*, 2010, pp. V232-V238.

- [12] J. Barksdale, E. Ragan and D. McCrickard, "Easing team politics in agile usability: A concept mapping approach", in proceedings of the Agile Conference, 2009, pp. 19-25.
- [13] D. D. Brown, "Five Agile UX Myths", *J. Usability Studies*, vol. 8, n° 3, 2013, pp. 55-60.
- [14] M. Seyam, "Enhancing usability through agility: Pair programming for a practice-oriented integration approach", in proceedings of the 2015 International Conference on Collaboration Technologies and Systems (CTS), 2015, pp. 460-463.
- [15] A. Bertholdo, T. Da Silva, C. De O. Melo, F. Kon and M. Silveira, "Agile usability patterns for UCD early stages", *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 8517 LNCS, n° 1, 2014, pp. 33-44.
- [16] D. Broschinsky and L. Baker, "Using Persona with XP at LANDesk Software, an Avocent Company", in proceedings of the Agile Conference, 2008, pp. 543-548.
- [17] V. Grigoreanu and M. Mohanna, "Informal Cognitive Walkthrough (ICW): Paring down and pairing up for an agile world", in proceedings of the Human Factors in Computing Systems, 2013, pp. 3093-3096.
- [18] P. Hodgetts, "Experiences integrating sophisticated user experience design practices into Agile processes", in proceedings of the Agile Conference, 2005, pp. 235-242.
- [19] M. Brhel, H. Meth, A. Maedche and K. Werder, "Exploring principles of user-centered agile software development: A literature review", *Information and Software Technology*, vol. 61, 2015, pp. 163-181.
- [20] T. Da Silva, A. Martin, F. Maurer and M. Silveira, "User-centered design and agile methods: A systematic review", in proceedings of the Agile Conference, 2011, pp. 77-86.
- [21] J. Ferreira, J. Noble and R. Biddle, "Interaction designers on eXtreme Programming teams: Two case studies from the real world", in proceedings of the 5th New Zealand Computer Science Research Student Conference, 2007, pp. 1-8.
- [22] Z. Hussain, H. Milchrahm, S. Shahzad, W. Slany, M. Tscheligi and P. Wolkerstorfer, "Integration of extreme programming and user-centered design: Lessons learned", *Lecture Notes in Business Information Processing*, vol. 31 LNBIP, 2009, pp. 174-179.
- [23] K. Petersen, R. Feldt, S. Mujtaba and M. Mattsson, "Systematic Mapping Studies in Software Engineering", in proceedings of the 12th International Conference on Evaluation and Assessment in Software Engineering, 2008, pp. 71-80.
- [24] X. Ferré, Marco de Integración de la Usabilidad en el Proceso de Desarrollo Software, 2005.
- [25] O. Sohaib and K. Khan, "Incorporating discount usability in extreme programming", *International Journal of Software Engineering and its Applications*, vol. 5, n° 1, 2011, pp. 51-62.
- [26] Å. Cajander, M. Larusdottir and J. Gulliksen, "Existing but not explicit - The user perspective in scrum projects in practice", *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 8119 LNCS, n° 3, 2013, pp. 762-779.
- [27] L. Caballero, A. Moreno and A. Seffah, "Persona as a Tool to Involving Human in Agile Methods: Contributions from HCI and Marketing", *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* 5th IFIP WG 13.2 International Conference, HCSE 2014, vol. 8742, 2014, pp. 283-290.
- [28] B. Losada, M. Urretavizcaya, J.-M. López and I. Fernández-Castro, "Applying usability engineering in InterMod agile development methodology. A case study in a mobile application", *Journal of Universal Computer Science*, vol. 19, n° 8, 2013, pp. 1046-1065.
- [29] B. Losada, M. Urretavizcaya, J.-M. López-Gil and I. Fernández-Castro, "Combining InterMod agile methodology with usability engineering in a mobile application development", in proceedings of the ACM International Conference, 2012, pp. 39:1-39:8.
- [30] H. Beyer, K. Holtzblatt and L. Baker, "An agile customer-centered method: Rapid contextual design", *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 3134, 2004, pp. 50-59.
- [31] W. Hudson, "Adopting User-Centered Design within an Agile Process: A Conversation", *Cutter IT Journal*, vol. 16, n° 10, 2003, pp. 5-12.
- [32] Adikari, C. McDonald and J. Campbell, "Little design up-front: A design science approach to integrating usability into agile requirements engineering", *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 5610 LNCS, n° 1, 2009, pp. 549-558.
- [33] M. Najafi and L. Toyoshiba, "Two case studies of user experience design and agile development", in proceedings of the Agile Conference, 2008, pp. 531-536S.
- [34] M. Düchting, D. Zimmermann and K. Nebe, "Incorporating user centered requirement engineering into agile software development", *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 4550 LNCS, n° 1, 2007, pp. 58-67.
- [35] W. Isa, A. Lokman, S. Aris, M. Aziz, J. Taslim, M. Manaf and R. Sulaiman, "Engineering rural informatics using agile user centered design", in proceedings of the 2nd International Conference on Information and Communication Technology, 2014, pp. 367-372.
- [36] W. Rahim, W. Isa, A. Lokman, N. Taharim and N. Wahid, "Engineering m-learning using agile user-centered design", in proceedings of the 8th International Conference on Next Generation Mobile Applications, Services and Technologies, 2014, pp. 60-65.
- [37] J. Haikara, "Usability in agile software development: Extending the interaction design process with personas approach", *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 4536 LNCS, 2007, pp. 153-156.
- [38] J. Kollmann, H. Sharp and A. Blandford, "The importance of identity and vision to user experience designers on agile projects", in proceedings of the Agile Conference, 2009, pp. 11-18.
- [39] Y. Nakao, M. Moriguchi and H. Noda, "Using agile software development methods to support human-centered design", *NEC Technical Journal*, vol. 8, n° 3, 2014, pp. 37-40.
- [40] P. Wolkerstorfer, M. Tscheligi, R. Sefelin, H. Milchrahm, Z. Hussain, M. Lechner and S. Shahzad, "Probing an agile usability process", in proceedings of the Conference on Human Factors in Computing Systems, 2008, pp. 2151-2157.
- [41] G. Meszaro and J. Aston, "Adding usability testing to an agile project", in proceedings of the Agile Conference, 2006, pp. 289-294.
- [42] T. Da Silva, M. Silveira and F. Maurer, "Usability evaluation practices within agile development", in proceedings of the Annual Hawaii International Conference on System Sciences, 2015, pp. 5133-5142.
- [43] A. Moreno and A. Yagüe, "Agile user stories enriched with usability", *Lecture Notes in Business Information Processing*, vol. 111 LNBIP, 2012, pp. 168-176..

Towards Optimization of Energy Consumption of Drones with Software-Based Flight Analysis

Luis Corral
ITESM / UAQ
Queretaro, Mexico
lrcorralv@itesm.mx

Ilenia Fronza, Nabil El Ioini, Aristeia Ibershimi
Free University of Bozen/Bolzano
Bolzano, Italy
{ilenia.fronza, nabil.elioni, aristeia.ibershimi}@unibz.it

Abstract—The capabilities and applications of new drones require a high demand of power that traditional batteries cannot sustain, triggering the need of developing strategies to promote efficient energy usage on drone platforms. However, to reduce the energy consumption it is required to have reliable means to measure the device's behavior and its relationship to the battery discharge. This paper introduces Green Flight, a software system that acquires data during a drone mission, featuring an online battery discharge analyzer and a real-time suggestions mode to help to optimize the overall power consumption of the drone, allowing for longer, more autonomous missions.

Keywords- *airworthiness, battery, drone, energy, power;*

I. INTRODUCTION

The fast development and diffusion of the new generation of unmanned aerial vehicles, more popularly known as drones, has brought a number of needs and opportunities for different research paths. The drone industry has been a hot topic for the past several years, and the focus of many discussions in the technology sector in different aspects: applications, handling, management, airworthiness, autonomy, and energy efficiency. In this regard, the capabilities of new generation drones boost a high demand of power that traditional batteries cannot sustain. This situation triggers the need of strategies to promote efficient energy usage to reduce the power demand on drone platforms, with the goal of guaranteeing the capacity of the drone to complete a mission safely. In the past, several research and practitioner work has focused towards designing autonomous, energy efficient targets (for instance, smartphones, tablets, or wearables) [1]; however, there is little research and experience applying these approaches in drone platforms. Moreover, an important need that arises when designing energy aware systems is the ability to understand the way in which the energy is used and spent. This opens the doors to design, implement and validate reliable means to measure the battery discharge for drone targets.

In this paper, we outline a strategy to measure the energy consumption of commercial quad drones. Our strategy includes a systematic data collection of drone missions, and online data analysis to characterize and report out the actions that relate the behavior of the drone to the overall consumption of energy. In this way, users may identify what are the maneuvers, movements and general aspects of the flight profile (load, altitude, speed) that can involve more energy investment, and as a consequence faster battery discharge that leads to shorter

drone autonomy. This knowledge can be of utmost importance for strategic actions, such as: identifying and monitoring power-relevant characteristics of drones; proposing metrics to evaluate the energy performance of the drone; and establishing baselines to manage energy consumption and identify energy relevant faults.

The rest of the paper is organized as follows: Section II covers the related work on energy aware software design for non-stationary platforms; Section III shows the project environment and solution approach; Section IV introduces Green Flight, a tool that acquires and reports drone flight data; Section V discusses the outcome of a series of executions in a real drone platform; Section VI sets directions for future work and draws conclusions.

II. RELATED WORK

The analysis of the state of the art in the field of non-stationary devices shows that the existing strategies are mostly based on common, autonomous devices such as smartphones, tablets or wearables. As battery technology and other hardware-based innovations run at a slow pace, a common approach is the design and implementation of software-based energy aware techniques [2]. These techniques include strategies to save energy at operating system level, implementing extensions for economic profiles [3], distributing the computing power to stationary resources [4], and in general, trying to reduce the energy footprint of software executed away from a power-loading source [5].

Many of these strategies have been implemented and validated in targets like smartphones and tablets. Nonetheless, just like smartphones or wearables, drones have to rely on their battery capacity since it represents one of the most important limitations to their operation; similarly, drones have to perform much of their function far from a stationary power-loading source. However, this constraint becomes much more relevant as a key operative attribute is airworthiness, which includes “sufficient power to maintain movement, to implement the controls, and to operate sensors and data-feeds, for the duration of the flight” [6]. Moreover, the potential usefulness of drones in many mission critical applications (such as emergency relief, surveillance, defense) is prevented by the battery life constrain. This limitation is increased in those cases where power resources are needed not only to fly successfully, but also to power on-board facilities such as a built-in videocamera.

III. PROJECT ENVIRONMENT AND APPROACH

Drones are aircrafts that have no onboard, human pilot. The term “remote pilot” is commonly used because the person handling the device is at a certain distance, but in full control of it. The object of this experimentation is “small drones”, since this kind of target is both energetically limited and technologically simple; moreover, small drones are commercially available for any user, so improvements on the aspects of energy consumption and increased autonomy may benefit a large number of users.

Small drones are subject to a range of limitations in relation to load capacity, flight duration (maximum autonomy of about 15 min), power supply, and other technical features. One of the most important limitations is the high power consumption which influences the flight duration of the drone mission. The power demand of drones is satisfied commonly using a battery, but the increasing intake of power and high requirement of autonomy on drones increases the need of a strategy towards efficient power consumption. The operational standard of drones should be aware and apply the most important quality requirement: *airworthiness*. This term is commonly used to refer to an aircraft's suitability for safe flight. A drone is considered airworthy if it is able to take off, conduct its mission and land safely [7]. Said this, *energy efficiency* towards *airworthiness* is the most important quality requirement to design a supporting solution.

A. Proposed approach

The main goal of this work is to design and build a software-driven solution that contributes to improve the energy consumption of drones by establishing tools, measurements and reports. This agenda was based in such way as to complete several tasks that we identified as “critical operations” to assess the energy status of the device, and provide a factual basis for suggesting strategies for energy consumption. These tasks are measurement and health management. As result, we developed a software system that allows to pilot a commercial small drone, which includes a data collection engine and a “real-time suggestions mode” that can be activated in order to optimize the overall power consumption of the drone. The suggestions provided are based on the data collected by the measurement engine, which tracks an approximation of the energy consumed by the execution of each maneuver during the flight, based on the readings of the battery level of the drone.

To achieve this goal, the first step is having the means for measuring the power consumption of the device. To handle this aspect, we designed a software tool which aims to measure with high granularity the power spent by the drone while it is flying, and to characterize it to the actions performed by the drone. The second step is to analyze the data collected acquired from the software tool, aiming to get some prognostics in order to manage the battery health. The third step is to use the analyzed data to build a software application with two main functionalities: (a) piloting the drone and show the current battery percentage as the flight goes on; and (2) activating the “real-time suggestions mode” (that is, a battery saving mode) once a critical battery percentage is reached. This mode consists of suggestions for the user in order to take fullest advantage of the remaining flight time.

This set of functionalities meet the critical operations identified above: measurement and health management. In summary, the software system, named “Green Flight”, monitors the action of the drone, measures the consumed battery and the overall flight time while the drone is flying, delivers data about how the drone is consuming the energy and permits future analysis in the form of suggestions for the user.

B. Proposed Evaluation and Selected Target

To assess the effectiveness of the system, it is necessary to implement and evaluate the system in a real target, and to conduct a series of experiments that let us understand the impact of Green Flight in the accomplishment of our goal. To this end, we considered two different aspects of evaluation: the first stage of the evaluation should assess the measurement tool created to track battery consumption of each maneuver during the flight, while the second stage evaluated the impact of the “real-time suggestion mode” on the overall battery life.

The drone used for implementing Green Flight and collecting data throughout the necessary experiments is a Parrot Rolling Spider¹. This drone is designed and commercialized for recreational and entertainment use only. The Rolling Spider is controlled by a smartphone or tablet that supports Bluetooth Low Energy (BLE). This means that the drone does not appear in the usual list of Bluetooth devices in smartphones, but it is only visible and able to connect to it through the host application [8]. The Rolling Spider weights 55 grams, and relies on a 550 mA, 1.1 V removable LiPo battery.

IV. SYSTEM DEVELOPMENT

A. System Design

Green Flight is implemented as an Android application which connects to the Parrot Rolling Spider drone via Bluetooth Low Energy (BLE). This application allows piloting the RS drone and keeping track of the drone battery consumption during the flight; moreover, the “real-time suggestions mode” is activated starting from 40% until 7% in order to optimize the overall battery consumption. Suggestions are based on data collected by a measurement tool that we created to track battery consumption of each maneuver during the flight. Green Flight is built using the Parrot ARSDK². As the topic is still pretty new and innovations are continuously added, the Parrot's software development kit is not fully documented, and only some sample applications are provided on Github, which depend on the type of Parrot Drone and OS of the piloting device. Since the target is a Parrot Rolling Spider drone, Green Flight was built by customizing one of the sample Android applications that let pilot the device. The customizations included functionality to collect data about flights and battery usage.

B. System Implementation

The user interface displaying the piloting part of the Green Flight application is placed in the lower half of the screen of an Android-enabled tablet. In the upper half of the screen, suggestions will be displayed during the flight.

¹ <http://www.parrot.com/usa/products/rolling-spider>

² <https://github.com/Parrot-Developers/ARSDKBuildUtils>

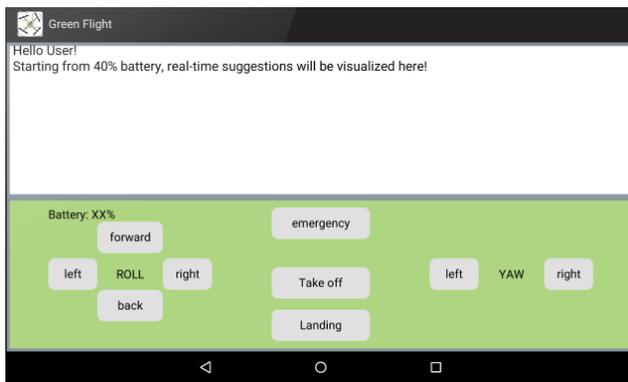


Figure 1: Green Flight GUI and available features.

The operations permitted to pilot the drone are: taking off and landing; rolling left, right, forward and backward; and yawing, that is, rotating either to the left or to the right side (Figure 1).

The first customization implemented permits to keep track of the battery consumption and characterizing it to the drone's movements. This customization measures the energy spent by the drone and associates this consumption to the maneuver performed and to its duration. In order to improve accuracy, during the flight, whenever a button (which commands a maneuver) is pressed, the event is logged together with the current battery state and current timestamp. The same happens even when the button is released, which means that the maneuver is completed.

The second customization is the implementation of the "real-time suggestions mode". This feature is inspired by the battery saver mode of smartphones. It is known that, while making use of a smartphone there are certain activities that consume more energy than others: Wi-fi and Bluetooth connection, screen brightness, and Global Positioning System (GPS) are included among the most common battery drainer actions [9]. In order to optimize battery consumption, many manufactures provide the so-called power-saving mode which manages for the user the most power-exhausting features of the smartphone. This mode turns off almost all the features except the necessary ones for making and receiving phone calls and sending text messages. For drones, there is no such a knowledge.

As an effective approach towards drone energy consumption optimization, in this work we proposed a mode where some suggestions of maneuvers are given while the operator is piloting the drone. These suggestions include maneuvers that are still *safe* to perform even with low battery levels (for example: "suggested to land", "suggested to move down, and so on). Maneuvers that are regarded by the system as "energy hungry" are never suggested, as performing them will have a serious impact in the already low battery level.

The knowledge to produce the suggestions is based upon flights analysis (the logs of the drone flight history), whose output identifies the most energy consuming drone maneuvers. To this end, the "real-time suggestions mode" uses the data collected by the first customization, that is, the energy measurement feature.

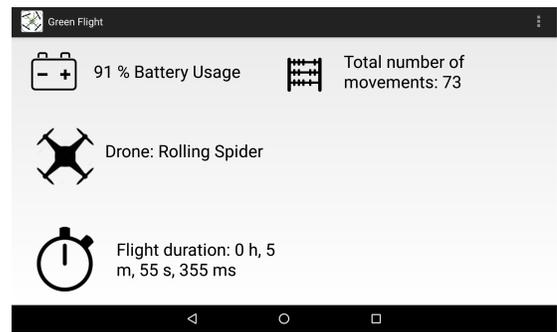


Figure 2: List of drone flights and summary of information.

An additional feature was implemented: "My Piloting". This functionality consists in a list of all the flights performed by the user using a specific device. The content of the event log file is necessary to retrieve information in order to calculate the flight duration, the number of maneuvers performed during a single flight, and the battery usage. With this information, "My Piloting" is populated with a list of flights, and when a flight is selected, the above mentioned information are displayed in detail (Figure 2).

To roll out the application, tests were performed in two different environments: indoors (in an empty classroom) and outdoors (in a cleared University court). The implementation of the Android application permitted obtaining full control of the device and operating it using the features provided in the GUI. By reaching a battery level threshold (40% of remaining battery power), the real-time suggestions are visualized.

V. SYSTEM EVALUATION

Green Flight proved its ability to track in real time all the performed maneuvers, and the number of times they were executed throughout the flight's timespan. Test flights served as a validation of the customizations implemented. The validation of the customizations was successful as: (1) Green Flight allows to pilot the drone properly, (2) the "real-time suggestions mode" is launched and is operational, and (3) at the end of each flight, its details were saved and added to a list of flights accessible through "My Piloting". Additional experiments were run to evaluate the measurement tool created to track battery consumption of each maneuver, and to evaluate the impact of the real-time suggestion mode on the battery life.

A. Evaluation of the measurement tool

This evaluation has conducted 4 experiments of 4 combined movements: up-down, back-forward, roll left-roll right, yaw left-yaw right. All the performed experiments started with fully-charged battery and went on doing the combined movements (e.g., up - down) until the battery percentage went down to 3% (time when the drone automatically shuts down and forces landing). Data were collected as follows: (i) action (pressed/released), (ii) battery percentage (for action pressed and released) and (iii) time (for action pressed and released). Energy consumption was measured 75 times for each maneuver of the combination.

For each flight of the experiments, a log file was created. This evaluation has shown that (1) during all the experiments, the log file was generated successfully and their content was

generated while piloting the drone. At the end of the flight, the generated log file was retrieved and further analyzed; (2) the measurement tool did not interfere with the already existent code; (3) Green Flight performed normally without crashing or losing the piloting control. Therefore, Green Flight can be defined as a tool that does not interfere with or diminish the conditions of airworthiness.

B. Evaluation of the suggestions for optimizations

After having implemented the “real-time suggestion mode”, the next step is to evaluate its impact to the battery consumption and drone’s lifetime. In order to perform this evaluation, there were performed four experiments without the suggestion mode of the Green Flight and other four experiments following the provided suggestions of the Green Flight. Considering all 8 experiments performed with and without the suggestion mode of the Green Flight, we present a summary of the experiments without (Table 1), and with suggestion mode (Table 2).

Table 1: Summary of 4 flights without suggestion mode.

Experiment	Duration of Flight
1	5 minutes, 44 seconds
2	5 minutes, 50 seconds
3	6 minutes, 5 seconds
4	6 minutes, 2 seconds

Table 2: Summary of 4 flights with suggestion mode.

Experiment	Duration of Flight
1	5 minutes, 53 seconds
2	5 minutes, 58 seconds
3	6 minutes, 3 seconds
4	6 minutes, 9 seconds

From this dataset, the boxplot in Figure 3 compares the flights with and without suggestions. We conclude that following the suggestions, the mean duration of the flight increases on about 5,6 seconds, showing a slight but noticeable contribution to the overall drone autonomy.

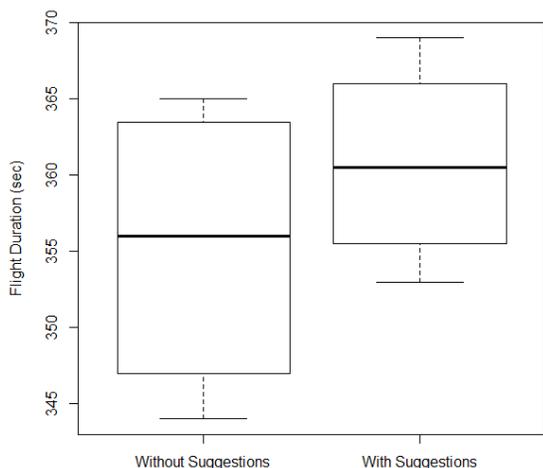


Figure 3: Box plot of the flight duration in seconds, with and without suggestions.

VI. FUTURE WORK AND CONCLUSIONS

In this paper we show the first software-based approach that aims to reduce the energy consumption of commercial quad drones: Green Flight, an Android application built on top of the Parrot ARSDK3. This software application has been evaluated by performing a series of experiments in a real target, the Parrot Rolling Spider drone.

An important constraint in the implementation of this project was imposed by the software development kit of the Parrot platform, since it is not well documented, and it does not allow APIs to access to the drone hardware. Without these constraints, we could have been able to implement a more accurate solution. This is an open issue that can be solved by new releases of the SDK, and by research works that follow up the present one. For reproducibility purposes, Green Flight should be tested using another kind of drone to strengthen the meaningfulness of the results obtained. Finally, a next step in the research roadmap is to use the data collected by Green Flight to develop tools that leverage this knowledge to suggest economic modes to be automatically inducted to the drone.

With the results obtained, we can conclude that having a tool that collects in real time relevant data from the drones, enables the user to identify the behavioral patterns that spend more energy, and as consequence, supply the necessary intelligence to avoid energy expensive maneuvers, leading to an operation oriented toward efficient energy consumption, illustrated in the real-time suggestion mode. By having this mode at hand, operators can extend the duration of missions, be aware of risky conditions created by the lack of battery, and promote the use of drones on critical missions or in scenarios that demand extended autonomy and high dependability.

REFERENCES

- [1] Procaccianti, G.; (2015) The green lab: Experimentation in software energy efficiency. *Proceedings of the International Conference on Software Engineering*, 2015. ACM.
- [2] Bornholt, J.; Mytkowicz, T.; & McKinley, K. S.; (2012) The model is not enough: Understanding energy consumption in mobile devices, *Power (watts)*, vol. 1, no. 2, p. 3.
- [3] Corral, L.; Georgiev, A.B.; Janes, A.; & Kofler, S.; (2015) Energy-Aware Performance Evaluation of Android Custom Kernels. *4th International Workshop on Green and Sustainable Software in connection with ICSE 2015*. pp. 1-7. ACM.
- [4] Lewis, G.; and Lago, P.; (2015) Architectural tactics for cyber-foraging: Results of a systematic literature review. *Journal of Systems and Software*. vol. 107 pp. 158-186. Elsevier.
- [5] Balan, R. K., Satyanarayanan, M., Park, S. Y., & Okoshi, T. (2003). Tactics-based remote execution for mobile computing. *Proceedings of the 1st international conference on Mobile systems, applications and services* (pp. 273-286). ACM..
- [6] Clarke, R.; (2014) Understanding the drone epidemic. *Computer Law & Security Review*, vol. 30, pp. 230-246, June 2014.
- [7] Corral, L.; Fronza, I.; & El Ioini, N.; (2015) The future of energy-aware software: The case of drones. *Cutter IT Journal, issue "What is Over the Technology Horizon"*. vol. 28 (8). pp. 19-23. Cutter Consortium.
- [8] Parrot S.A.; (n.d.) Rolling Spider User Guide UK. www.parrot.com/support/parrot-rolling-spider. Retrieved Feb 5th, 2015.
- [9] Corral, L.; Georgiev A.B.; Sillitti, A.; & Succi, G.; (2013) A method for characterizing energy consumption in Android smartphones and tablets. *2nd International Workshop on Green and Sustainable Software in connection with ICSE 2013*. pp. 38-45. ACM.

Choosing the Best Strategy for Energy Aware Building System: an SVM-based Approach

Yuanyang Wang, Xiaohong Chen*, Haiying Sun, Mingsong Chen

Shanghai Key Laboratory of Trustworthy Computing, East China Normal University, Shanghai, 200062, China

*corresponding author. xhchen@sei.ecnu.edu.cn

Abstract—For many old buildings in the world, due to the legacy devices problem, it is hard to supply appropriate energy for them. In order to reduce the energy consumption of buildings under the premise of satisfying user requirements, we use software control systems whose core part is the scheduling strategy, to reconstruct them. It is time consuming to choose a good scheduling strategy due to many uncertain factors, among which user actions are of the most influence. In this paper, we propose an Support Vector Machine (SVM) based approach to explore the relation between user action and the best scheduling strategy of a control system. The main contributions include: (1) obtaining the sample set by collecting data at the model level using Statistical Model Checking (SMC) based method; (2) using SVM algorithm to learn the relation model between user actions and the best scheduling strategies; and (3) applying the relation model to predict a best scheduling strategy. Finally a real case study is conducted showing the efficiency of our approach.

Keywords—Energy Aware Building; User Actions; Scheduling Strategy; Support Vector Machine; Statistical Model Checking.

I. INTRODUCTION

Although there are many new buildings which embed intellectual technology to minimize energy consumption, there are still a huge number of old buildings left. They do not have sensors, or enough devices, i.e., heaters to supply energy for each room at the same time. It is difficult to manually schedule heaters of the whole building to satisfy every user with least energy consumption. To financially reconstruct these buildings for minimizing energy consumption, designing a software system which provides a scheduling strategy to schedule heaters and control the energy consumption of the whole building is a good choice.

In order to design such a system easily for the system designer, we make the stable part of the software system to be a *Controller*, and the changeable part a *Scheduling strategy* [1]. Many factors should be considered when designing a scheduling strategy, for example, the rooms, the users and the weather [1] [2]. Among all these factors, we argue the users factor is a very important one because the user actions such as arriving the building and leaving the building finally determine the total energy the users need. But the actions of users are difficult to study, because they are uncertain, which means they are changing and can not be controlled. Designers can only monitor them, and use monitoring results to design proper scheduling strategies. To facilitate the system designer, we propose to define a few scheduling strategies beforehand. The designers could choose one from them such as the one that consumes the least energy. Our previous work

presents an Statistic Model Checking (SMC) based framework to evaluate the energy consumption [1] of each strategy. In that framework, user action is also one factor that is modelled by Stochastic Hybrid Automata (SHA). The energy evaluation is conducted by the probabilistic model checker UPPAAL-SMC [3]. However, each time the evaluation takes about 2.5 hours. Given 100 strategies, to choose one needs 250 hours. How to quickly choose a best strategy which consumes the least energy under the premise of satisfying user requirements, if only the users are changed, is an important problem that needs solving.

In this paper we intend to explore the relation between user actions and the scheduling strategy chosen in order to improve the efficiency of design procedure. The machine learning based approaches [4] give us the inspiration that we do not need to do the evaluation every time. We could select some samples, and learn the relation using machine learning algorithms. We regard the best strategy choosing problem as a classification problem as the user actions which have the same best scheduling strategy could be divided into the same class. Therefore, in this paper we propose an Support Vector Machine (SVM) [5] based approach to learn the relation between user actions and the best scheduling strategy, and use the relation learnt to predict the best scheduling strategy for specific building automatically. The sample set is acquired by applying our SMC based evaluation framework [1]. This is also because old buildings usually do not have any sensors to monitor its own running status. When we do the application, the user actions in terms of arriving and leaving a building are expressed quantitatively by data fitting tool of MATLAB [6].

The rest of the paper is organized as follows. Section II briefly introduces the SMC-based evaluation framework. Section III presents our approach. Section IV conducts a real case study, and finally section V concludes the paper, and put forwards our future work.

II. INTRODUCTION TO SMC-BASED EVALUATION FRAMEWORK

In our previous work [1], the SMC based evaluation framework includes two parts, i.e., the system modeling, and the energy evaluation. The system is modelled by a Network of Stochastic Hybrid Automata (NSHA), while each component of the system is modelled by an SHA. Using the energy aware building system as an example, we introduce the system modeling especially the user modelling.

The energy aware building system is composed by *Controller* and four interactive components, i.e., *Weather*, *User*, *Room* and *Heater*, where *Controller* is the software to-be built. The scheduling strategy is embedded in the *Controller*. Each of them is modded by an SHA. Each SHA model has some parameters which may represent the domain knowledge such as the probability of rain, the power of heater and the room amount of a building. They need to be configured before getting a concrete system. In the following, we introduce the *User* model in detail for further use.

Fig. 1 shows the SHA model of *User*. There are four basic locations, *Start*, *Wait*, *In* and *Out*. They have a time attribute t_{day} which increases linearly as the time goes on. For each room, when the time is later than arrive time $arrive_t$, it transfers from *Wait* to *In*. When the time is later than leave time $leave_t$, it transfers from *In* to *Out*. The $arrive_t$ and $leave_t$ are allocated by the function $time_init()$. The parameters of *User* include the range and distribution of $arrive_t$ and $leave_t$.

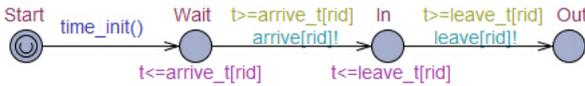


Fig. 1. *User* model SHA

After getting the concrete system, we conduct the energy evaluation on the probabilistic model checker UPPAAL-SMC [3] by property queries. Based on this, different scheduling strategies could be compared easily.

III. OUR APPROACH

The framework of our approach is shown in Fig.2. There are two parts, i.e., the SVM-based relation model training and checking, and the application. The former part uses SVM algorithm to train the relation model, and check the predicting accuracy of the model. The latter part is used to predict a best scheduling strategy for new user actions.

A. Relation model training and checking

The input of this part is the instances of the parameters of SHA models, and the output is a relation model. This part includes for steps: collecting original data, acquiring sample set, learning relation, and checking accuracy.

Step 1. Collecting original data

The original data means the user actions, and the best scheduling strategy at that time. This could be done by using the SMC-based evaluation framework. To be exact, this step includes three sub-steps, i.e., specific system configuration under certain situations, energy consumption evaluation, and original data extraction.

Firstly we conduct the system configuration by setting values to the parameters of *Weather*, *Room*, *Heater* and *Controller* according to domain knowledge. Here we focus on the *User* and *Controller*. The parameters $Arrive_range$ and $Leave_range$ are easy to set according to actual situation. The distribution of $arrive_t$ and $leave_t$ can be expressed by

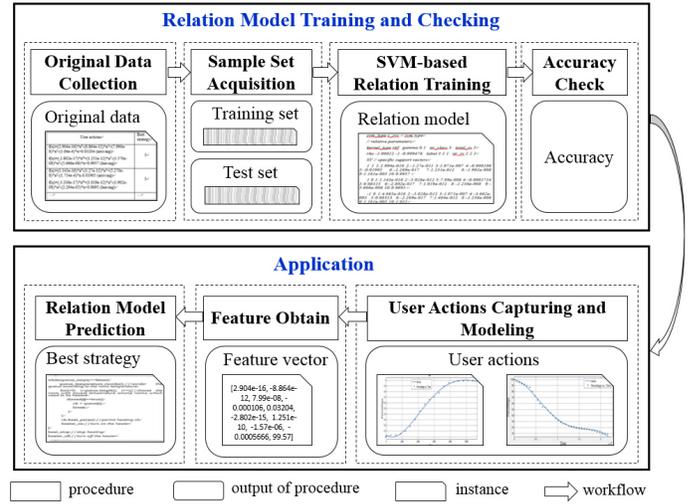


Fig. 2. The framework of our approach

the relation function between time and Percentage of Users (PU) in the building which can be calculated as formula (1). The data fitting result shows that polynomial function is suitable for being the fitting function. This is also reasonable in theory according to the characteristic of user actions.

$$PU = \frac{\text{Number of users in the building}}{\text{Number of total users}} \quad (1)$$

The controller configuration means to select a scheduling strategy from existing ones or design a new scheduling strategy for system according to user requirements.

Then we conduct the query $Pr[<= day](\langle \rangle energy >= 20000000)$ which intends to explore the probability of energy consumption exceed 20000000 energy units within a day. The best scheduling strategy can be obtained by directly comparing the cumulative probability of each strategy. The lower the cumulative probability is, the better the strategy is. The corresponding user actions and its best scheduling strategy is recorded. In order to get more original data, we need to find the best strategies under different user actions PU_A and PU_L while other configuration is fixed. We change the parameters of user action expressions randomly in a reasonable scope. Based on the evaluation framework, we get the best strategies under different user actions which called original data of our approach.

Step 2. Acquiring sample set

We extract the sample set from the original data. A sample is composed by features and a label. In our approach, the parameters of user action expressions are regarded as features. And the label is just the number of best scheduling strategy. For example, if the best strategy under some specific user actions is Strategy 2 which has been predefined, the label of this sample is just 2. The sample set is divided into two subsets: the training set and the test set. The training set is used by SVM algorithm to learn the relation model we need. And the test set is used to check accuracy of relation model predictions.

Step 3. Training relation model

Based on the training set, the relation between user actions and energy aware building system designs can be obtained by applying SVM algorithm. In this paper, we call the LIBSVM [7] to implement the SVM classification.

Step 4. Checking Accuracy

We use the test set to check the accuracy of relation model prediction results. Its defined as the percentage of correct predictions in total predictions as shown in formula (2). To some extent, the higher the accuracy, the better the relation model is.

$$Accuracy = \frac{\text{Number of correct predictions}}{\text{Number of total predictions}} \quad (2)$$

B. Application

The application uses the obtained relation model to predict the best scheduling strategy for specific to-be reconstructed building without modeling the whole building. The input of the application is the user actions of specific building and the obtained relation model, and the output is the predicted best scheduling of this building.

Step 1. Capturing and modeling user actions

The designers are expected to capture user actions at first. And then modeling them by data fitting tools in order to get user action expressions. Considering the convenience of extracting features, the user action expressions should be in the format required by the prediction model.

Step 2. Obtaining features

The parameter value of functions extracted from user actions will be regarded as the feature value of this sample. They will be stored in a feature vector which is regarded as the input of relation model.

Step 3. Predicting best strategy

After the feature vector obtained, software system designers just need to input it into relation model. The relation model will output the predicted best scheduling strategy automatically from given ones.

IV. CASE STUDY

We take the Science Building in East China Normal University (ECNU) as an example to show the efficiency of our approach, which is also used in [1]. We use the three predefined scheduling strategies in that paper, named Strategy 1, Strategy 2, and Strategy 3.

A. Relation model training and checking

1) *Collecting original data:* The system configuration is fixed (omitted due to the limited pages which can be found in [1] except for the user actions and strategy. We use the attendance data of Science Building in March of 2015 to get the distribution of $arrive_t$ and $leave_t$. Each parameters of fitted curve given by MATLAB has an interval. We select 100 kinds of user actions randomly to conduct the evaluation on UPPAAL-4.1.19. That is to say, we get 100 original data.

TABLE I
THE USER ACTIONS IN SCIENCE BUILDING: EXAMPLES

Time	Users	PU	Time	Users	PU
7:00	4	0.55%	16:00	733	100.00%
7:10	19	2.59%	16:20	699	95.36%
7:20	25	3.41%	16:40	674	91.95%
...
9:50	731	99.73%	21:40	70	9.55%
10:00	733	100.00%	22:00	67	9.14%

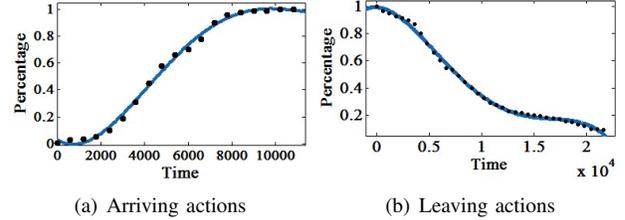


Fig. 3. User actions

2) *Acquiring sample set:* The sample set has 100 samples. We select 70 of them by stratified sampling method [8] to be the training data, and the remaining ones to be the test data.

3) *Training relation model:* We call the libsvm-3.20 with default parameters to training the 70 samples.

4) *Checking Accuracy:* The 30 test samples are used to test the prediction accuracy of relation model. The experiment result is that 26 predictions are correct. The accuracy is 86.7%. The testing results shows that our relation model is relatively good, and can be applied to practical use for predicting a best scheduling strategy under a specific kind of user actions.

B. Application

1) *Capturing and modeling user actions:* The attendance data of Science Building in April and May of 2015 is applied to conduct the case study. The data of each day is added together in order to obtain a more reliable distribution. For the arriving actions, we count the arrived users in each time interval which is 10 minutes here between 7:00 am and 10:00 am. As for the leaving actions of Science Building, we collect the number of leaving users every 20 minutes between 16:00 pm and 22:00 pm. Then we get the number of remained users by using number of users to subtract number of leaved users. The results are showed in Table I.

We use MATLAB7.1 to fit the curve of data in Table I (left). The time points are transformed into seconds. In order to simplify the calculation, we change the time 7:00 into 0, 7:10 into 600 as 10 minutes equals to 600 seconds. The rest can be done in the same manner. The result is shown in Fig.3(a). The X position is the time whose unit is second. The Y position is the PU in the building. The leaving action is processed in the same manner. And the result is shown in Fig.3(b). The expressions of user actions are shown in formula (3) (4) which correspond to Fig.3(a) and Fig.3(b) respectively.

$$PU_A = 2.904e - 16 * x^4 - 8.864e - 12 * x^3 + 7.99e - 08 * x^2 - 0.000106 * x + 0.03204 \quad (3)$$

$$\begin{aligned}
PU_L = & -2.802e-15 * x^4 + 1.251e-10 * x^3 \\
& -1.57e-6 * x^2 - 5.666e-6 * x + 0.9957
\end{aligned}
\tag{4}$$

2) *Obtaining features*: After the expressions of user actions obtained, we extract the parameters of user actions, and put them into a feature vector stored in a txt file shown as follows:[2.904e-16, -8.864e-12, 7.99e-08, -0.000106, 0.03204, -2.802e-15, 1.251e-10, -1.57e-06, -0.0005666, 0.9957] where they are values of parameter P_1 to P_{10} .

3) *Predicting best strategy*: And then transform it into the format can be accepted by LIBSVM. Later, we use the relation model acquired above to check this sample just by running the model using the feature vector as an input. For this case, the predicted best scheduling strategy is Strategy 3.

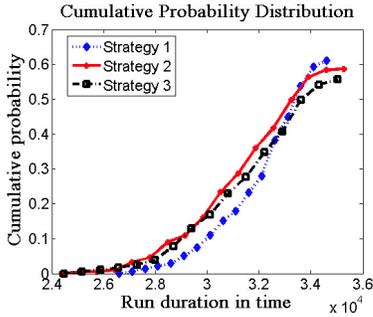


Fig. 4. Energy consumption cumulative probability of three strategies

C. Evaluation

In order to check whether Strategy 3 is the best strategy or not, we run the whole system model by importing the user actions. The simulation result given by query $Pr[\leq \text{day}](\langle \rangle \text{energy} \geq 20000000)$ in Fig.4 shows that the cumulative probability of Strategy 3 is smallest. It means that the probability of energy consumption exceed 20000000 energy units within a day is smallest by using Strategy 3. The application result shows that our approach can correctly predict the best scheduling strategy for specific building. It will greatly reduce the workload of building system designers.

In this application, the average energy consumption under each scheduling strategy of Science Building is shown in Table II. We can see that the Strategy 3 is the best strategy with the lowest energy consumption, and the Strategy 1 is the worst one. If the designers choose Strategy 1 or Strategy 2 as the scheduling strategy unfortunately, the energy consumption will increase by 2100000 or 800000 energy units each day respectively. Maybe the value for one day is not very big. But for one month, one year, the value will be huge. These results show the effectiveness of our approach. They demonstrate that using the scheduling strategy our relation model recommended can reduce the building energy consumption effectively without modeling the whole system or evaluating each strategies. Assuming that there are 100 buildings need to be allocated scheduling strategy from three ones. Our approach can give out the predicted strategy automatically. We just

TABLE II
ANALYSIS OF A SIMULATION

Strategy	Energy consumption	Analysis	Energy saving
1	2.19e7	Worst	2100000
2	2.06e7	-	800000
3	1.98e7	Best	-

need a little time to model the user actions. Without our approach, you approximately need $2.5*3*100=750$ hours to allocate scheduling strategies for the 100 buildings.

V. CONCLUSIONS

For old buildings reconstruction, choosing a good scheduling strategy for limited heaters is important for reducing energy consumption. We regard this issue as a classification problem in machine learning, and propose to use the SVM algorithm to learn a relation model between between user actions and best scheduling strategies. The SVM sample set is collected on the model level by applying an energy consumption evaluation framework.

Our experiment based on the data collected from Science Building of ECNU shows that our approach can correctly and quickly select a best scheduling strategy within a short time. Using the strategy recommended by our relation model can reduce considerable energy consumption in our application example. These all show the correctness and effectiveness of our approach. Of course, this is only an initial work. There are still much work left. For example, we should add more samples to improve the predicting accuracy, and add more aspects to measure the best scheduling strategy such as the comfort of users.

ACKNOWLEDGMENT

This work was supported financially by the National Natural Science Foundation of China for Young Scholars (Grant No. 61202104, 91418203), and the Doctoral Fund of Ministry of Education of China (Grant No.20120076120016).

REFERENCES

- [1] X. Chen, F. Gu, M. Chen, D. Du, J. Liu, and H. Sun, "Evaluating energy consumption for cyber-physical energy system: an environment ontology-based approach," in *Computer Software and Applications Conference (COMPSAC), 2015 IEEE 39th Annual*, vol. 2. IEEE, 2015, pp. 5–14.
- [2] A. David, D. Du, K. G. Larsen, M. Mikučionis, and A. Skou, "An evaluation framework for energy aware buildings using statistical model checking," *Science China information sciences*, vol. 55, no. 12, pp. 2694–2707, 2012.
- [3] P. Bulychev, A. David, K. G. Larsen, A. Legay, M. Mikučionis, and D. B. Poulsen, "Checking and distributing statistical model checking," in *NASA Formal Methods*. Springer, 2012, pp. 449–463.
- [4] L. Wu, G. Kaiser, D. Solomon, R. Winter, A. Boulanger, and R. Anderson, "Improving efficiency and reliability of building systems using machine learning and automated online evaluation," in *Systems, Applications and Technology Conference (LISAT)*. IEEE, 2012, pp. 1–6.
- [5] C. Cortes and V. Vapnik, "Support-vector networks," *Machine learning*, vol. 20, no. 3, pp. 273–297, 1995.
- [6] "Matlab," <http://www.mathwork.com>.
- [7] "Libsvm," <http://www.csie.ntu.edu.tw/~cjlin/libsvm>.
- [8] P. S. Levy and S. Lemeshow, *Sampling of populations: methods and applications*. John Wiley & Sons, 2013.

An Adaptable Approach for Indoor Location

Mário Andrade Vieira de Melo Neto
Federal Institute of Rio Grande
do Norte – Natal, Brazil
Email: mario.melo@ifrn.edu.br

Gibeon Soares de Aquino Júnior
Federal University of Rio Grande
do Norte – Natal, Brazil
Email: gibeon@dimap.ufrn.br

Abstract—In general people often spend 80-90% of their time in indoor environments, which include shopping malls, libraries, airports, universities, schools, offices, factories, hospitals, among others. In these environments, GPS does not work properly, causing inaccurate positioning. Currently, when performing the location of people or objects in indoor environments, no single technology can reproduce the same results achieved by the GPS for outdoor environments. One of the main reasons for this is the high complexity of indoor environments where, unlike outdoor spaces, there is a series of obstacles such as walls, equipment and even people. Due to this, it is necessary to consider the use of information from multiple sources using different technologies. Thus, this work proposes an adaptable approach for indoor location, which allows the use and combination of different technologies, techniques and methods in this context.

I. INTRODUCTION

Nowadays, it is clear that location systems are increasingly present in people's lives. These systems can help people solve different kinds of problems in a great variety of situations. People in general often spend 80-90% of their time in indoor environments [10], which include shopping malls, libraries, airports, universities, schools, offices, factories, hospitals, among others. Because of this, services that allow location in indoor environments have been gaining special attention. One of the reasons for the increase in popularity for this type of application is the popularization of portable devices such as cell phones, smartphones, PDAs, tablets and notebooks, which already have many built-in hardware such as WiFi, Bluetooth, GPS and inertial sensors.

There has been an increase in the demand for the location of people or objects in indoor environments to be a reliable and accurate [4]. In this sense, large technology companies such as Google and Apple are currently investing in this research area in order to develop solutions for location in indoor environments. This shows that this problem remains unsolved, that is, there is still no technology or combination of technologies that can solve the problem in an acceptable manner and with low costs [5]. One of the main reasons for this is the high complexity of indoor environments where, unlike outdoor environments, there are different obstacles such as walls, equipment and even people [7].

Thus, it is necessary that the solutions proposed to solve the problem of location in indoor environments take into account the complexity of these environments. Melo and Aquino [6] showed that there is a tendency for these solutions to combine the use of different technologies, sources of information,

location techniques, among other features, which will allow the adaptation of the solutions to the various complexities that can be found in these environments. In order to come up with a solution that fits all of the environments and that can be adapted to their specific characteristics, this paper proposes an adaptable platform that enables the combination and use of many techniques and technologies in order to obtain the location of people or objects in indoor environments.

The paper is organized as follows: Section II will be dedicated to the presentation of the proposed platform, detailing its requirements, architecture and components. In Section III, the evaluation of the platform's behavior applied to the context of location in indoor environments using WIFI and RFID technologies is presented. Section IV presents some related works, comparing them and pointing out strengths and weaknesses. Finally, Section V presents the final considerations, as well as future studies for this work.

II. PLATFORM ARCHITECTURE

The proposed platform defines a number of components that may have their functionalities adapted in order to enable the reception, processing and storage of many heterogeneous data that make up an indoor environment. Furthermore, the possibility of the adaptation of its components allows the platform to be adapted to different indoor environments and enables it to meet the specificities found in each one of them. Hence, new approaches, algorithms or techniques can be added without the need to adjust the base architecture.

In Figure 1, the general architecture of the platform is presented. This platform has a predefined set of components which are called main components. The proposed platform has 8 main components which are: I/O Manager, Request Manager, Map Manager, Data Publication Manager, Location Manager, Event Manager, Things Manager and Data Storage Manager. Each one of these components provides an adapter interface (AI) so that they may have its functionalities adapted by the adapter components. Moreover, given the separation of the architecture in different components, it is possible to adapt specific components, which making it optional to extend all of the platform's components in order to include or modify the capabilities.

The process of adapting the functionalities of the main components are on each component allows the adaptation of its functionalities through the AI using the Whiteboard Pattern [3]. Instead of searching in a directory for the component that

implements the required functionality, the main component registers the interest in components that are able to implement the adaptation of its functionalities. So whenever an adapter component is registered in the component directory, it checks if there is any main component interested in it. If there is, it is notified. Upon receiving the notification, the main component has access to the registered component instance, which performs the adaptation of its functionality.

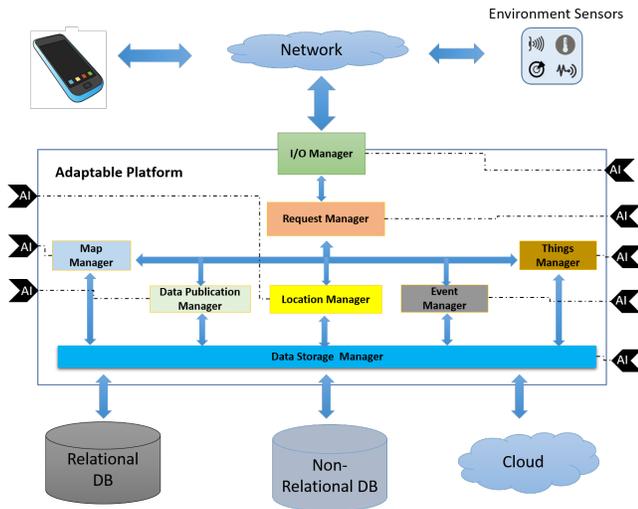


Fig. 1: Adaptable Platform Architecture Overview

A. I/O Manager

This is the component that handles the input and output towards the platform. The main goal of this component is to ensure that the platform enables the handling of heterogeneous data, allowing the input of any information regardless the technology or unit used in order to provide many types of location services.

B. Request Manager

This component acts as a dispatcher identifying the type of request sent to the platform, forwarding the request to the main component so it can perform its processing. This component can have its functionalities adapted. Thus, new requests for processing flow can be created, considering aspects not implemented by the main components.

C. Data Publication Manager

It is the component responsible for allowing the data produced on the platform to be accessed by client applications. This component will carry out the provision of information received and produced by the platform using the content providers made available by the *Data Storage Manager* component in order to provide this information.

D. Map Manager

If the request type is related to the maps domain, this is the component that will be responsible for receiving the request, transforming the information for this domain and invoking the service that will process the request.

E. Location Manager

This is the main component of the platform and is responsible for performing the processing of location information. This component's extensions must be able to execute three different behaviors. The first behavior is to turn the received information into data that can be processed by the component. The second identify the best strategy for processing this information. In this processing stage, it might use different location algorithms, a combination of them as well as single algorithm that can perform the fusion of information in order to generate new types of information. Finally, the received and processed information must be sent to storage so that other functionalities can use them.

F. Event Manager

This component will allow the use of the Push interaction mode [1] using the , in which it is possible to perform the registration of events by the implementation of the publish-subscribe pattern.

G. Things Manager

This component will be responsible for managing the platform, performing tasks such as registration, modification and removal of "things" that are necessary to platform operation. These "things" can be environments' equipment, devices, management data, among others.

H. Data Storage Manager

This component will serve as a data storage provider to the main components access the information persisted by the platform. This component support different forms of data storage, i.e., relational database, non-relational database, cloud storage services and others.

III. EVALUATION

In order to evaluate the proposed platform, an Android application was used to collect data from WIFI access points present in the given environment. In addition to these data, other data from RFID tags found in specific points of the environment were also collected. In every data collection, the data was sent to the platform, which processes it and calculate the estimated position of the device. In order to carry out this evaluation, the scenario used to for this is presented in the Figure 2. To execute this evaluation it was needed to create four adapter componentes which are: **RFID Service** – this component implements the functionality that processes data obtained from the RFID tags by the client applications, **WIFI Fusion Service** – performs the adaptation of the location functionality obtaining the user location using WIFI and RFID data, **WIFI Location Storage and RFID Location Storage** – These components are responsible for performing the storage of the WIFI and RFID data received by the platform.

The environment used for this assessment is a conventional office in a commercial building measuring 115 m². In this environment, there are four access points, so the existing infrastructure was used to perform this assessment. In addition,

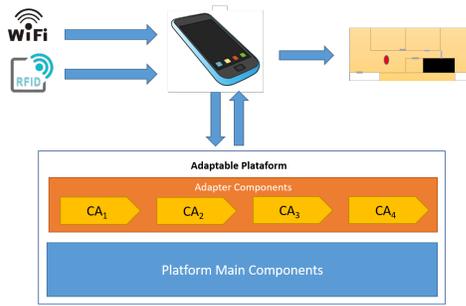


Fig. 2: Evaluation Scenario

there is one RFID tag attached to a wall in one of the rooms in the environment. Thus, a walking test was performed within the office and its result is shown in Figure 3. In the Figure, the environment map shows two routes. The dashed line represents the actual route, or the route that was taken by the user. The solid line represents the route calculated by the platform using the strategies defined by the adaptable components.

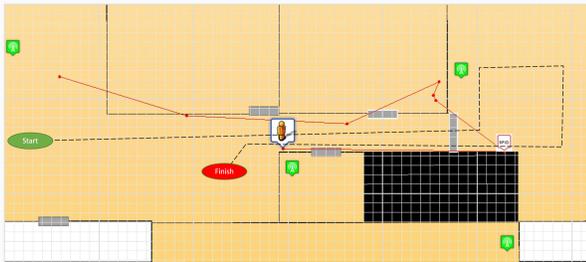


Fig. 3: Android app using the proposed platform for indoor location

The evaluation was performed under two facets: main components processing time compared to the total platform processing time and the easiness of the creation and incorporation of adapters components. In order to assess the processing time of the designed platform, the measurement of the time required for a request to be fully processed within it was performed. Thus, the time spent from the moment the request was received until the response was sent to the requestor was measured. The requests sent to the platform was to calculate the indoor location of the device. This request was chosen because is the most costly to the platform. The execution context in which the platform was implemented is an Intel Core i7 1.8 GHz machine with 8 GB of ram. However, for this assessment, the memory usage was limited to 512 MB. Accordingly, the average time that it takes a certain amount of requests sent at a time to be processed by the platform was calculated. The results are shown in Figure 4, in which the X axis represents the amount of requests sent and the Y axis represents the average processing time in milliseconds. In addition, for each amount of requests sent, ten samples were collected and their average time was calculated.

It is possible to note that the main component processing time does not exceed 0,11 milliseconds which represents

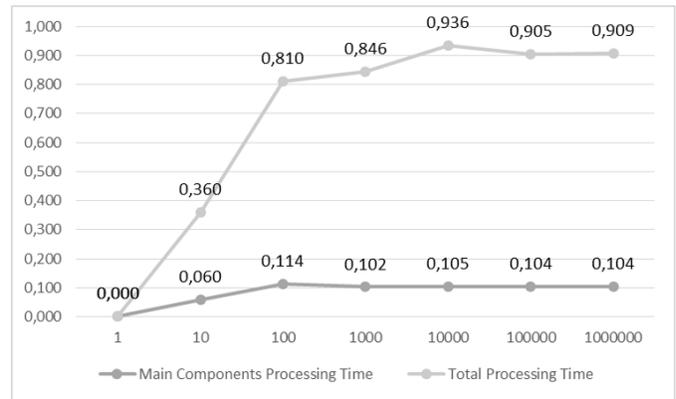


Fig. 4: Requests Processing Time

11% of the total time to processing the requests sent to the platform. This demonstrates that the greatest amount of average processing time is spent with the logical associated with the adapters components. Another fact that was possible to note is that the platform was able to keep the average processing time in stable values even increasing load 100 times higher.

TABLE I: NUMBER OF LINES OF CODE FOR THE EVALUATION

Component	Lines of code	Lines for adaptations
WiFi Fusion Service	157	16
WiFi Location Storage	127	9
RFID Service	160	16
RFID Location Storage	143	9
Total	587	50

Regarding the easiness to create and incorporate adaptations, we collected the amount of lines of code implemented in each of the adapter component, as shown in Table I. On this count, all of the lines of code in the source code's file were accounted for, including imports, statements, etc. In addition, we also counted the lines of code that are directly related to tasks that is necessary to incorporate these adapter components to the platform. Therefore, were required total 64 lines of code for the components to be coupled to the platform. Thus, it was necessary less than 9% of total lines of code to incorporate the adaptabilities the platform.

IV. RELATED WORK

In this section, related works directly associated with what was proposed in this work will be presented. We will highlight the main contributions of each of the listed works and will try to show the main differences between them and the proposed approach.

The Location Stack Architectural model introduced by Hightower et al. [2] shows a standardized model for developing location systems using a layer architectural model which refers to standard ways to perform the combination of data from different sources using a software architecture based on the Open System Interconnect (OSI) network communication

model. As the main difference between the Location Stack and the proposed platform's architecture, it is possible to note that the fact that it defines rigid layers - just like the OSI model in which a particular layer only receives data from the layer right below it and only exports the processed data to the layer right above it - makes the architecture only a little flexible. This little flexibility makes it hard to extend the functionalities, which differs greatly from the proposed architecture.

In Najib et al. [8], a middleware for location in indoor environments called MapUme was presented. The MapUme allows the fusion of data from various types of sensors. The architecture used is based in an architecture proposed by Hightower et al., in which it is organized in layers. Also, it uses the service-oriented architectural model. Despite being based on a modular architecture and divided into layers, this middleware allows the adaptability of its components. However, the adaptability is only allowed for some of the architecture's components. Even when the adaptability is allowed, it must occur with a high degree of coupling, since the new implementation must be built and coupled to the middleware. This feature makes it very different from the proposed platform, which allows the inclusion of new features without the need for interrupting the platform implementation.

In Ranganathan et al. [9], a middleware sensitive to location called "MiddleWhere" is proposed. This platform integrates multiple location technologies as well as presents the consolidated location data to customer applications. In order to do this, it uses a layered architecture to collect information from sensors and persist the information in a special database; an intelligence engine to perform data fusion in order to obtain the location information; and a service layer to make such information available. It also allows the pull and push interaction mode for communication with applications. Unlike the approach proposed in this paper, this middleware only allows a small degree of adaptability, making it only possible to vary the runtime of the insertion and removal of new sources of information. The proposed platform's architecture allows the use of new and different sources of information at runtime, as well as allows the modification of any of the behaviors found in the platform.

V. CONCLUSION AND FUTURE WORKS

This work presented the design, implementation and assessment of an adaptable platform for location of people and objects in indoor environments, which allows the use of different technologies, information sources, techniques, among other characteristics, in order to adapt itself to different and complex indoor environments. Furthermore, it was shown that the proposed platform is divided into main and adapter components. These main components are provided by the platform's architecture and can have their functionalities adapted by the adapter components.

The proposed platform was evaluated using an implementation in which WIFI data obtained from access points and RFID data obtained using tags were used. To collect such data an Android application was built. Furthermore, the platform were

evaluated under two aspects: main components processing time compared to the total platform processing time and the easiness of the creation and incorporation of adapters components. The main components processing time cause minor impact on the total process time, even with increasing amounts of requests sent. Regarding the easiness of creating and incorporating adaptability's, indicates to be an easy task. We realized that the amount of necessary lines of code on the specific parts of the platform up less than 10% of the total.

As future works, we intend to assess platform operation using other technologies and techniques in order to identify potential improvements in the proposed architecture. Furthermore, there is the intent to evaluate other characteristics related to platform implementation, such as processing and distributed scalability. Besides, we also intend to assess the use of cloud computing concepts and big data.

VI. ACKNOWLEDGMENT

This work was partially supported by the National Institute of Science and Technology for Software Engineering (INES) funded by CNPq under grant 573964/2008-4

REFERENCES

- [1] Manish Bhide, Pavan Deolasee, Amol Katkar, Ankur Panchbudhe, Krithi Ramamritham, and Prashant Shenoy. Adaptive push-pull: Disseminating dynamic web data. *Computers, IEEE Transactions on*, 51(6):652-668, 2002.
- [2] Jeffrey Hightower, B. Brumitt, and G. Borriello. The location stack: a layered model for location in ubiquitous computing. In *Mobile Computing Systems and Applications, 2002. Proceedings Fourth IEEE Workshop on*, pages 22-28, 2002.
- [3] Peter Kriens and B Hargrave. Listeners considered harmful: The whiteboard pattern. *Technical whitepaper, OSGi Alliance*, 2004.
- [4] F. Lemic, V. Handziski, N. Wirstrom, T. Van Haute, E. De Poorter, T. Voigt, and A. Wolisz. Web-based platform for evaluation of rf-based indoor localization algorithms. In *Communication Workshop (ICCW), 2015 IEEE International Conference on*, pages 834-840, 2015.
- [5] Dimitrios Lymberopoulos, Domenico Giustiniano, Vincent Lenders, Maurizio Rea, Andreas Marcaletti, et al. A realistic evaluation and comparison of indoor location technologies: Experiences and lessons learned. *ACM/IEEE International Conference on Information Processing in Sensor Networks*, pages 178-189, 2015.
- [6] Mário Melo and Gibeon Aquino. Categorization of technologies used for fingerprint-based indoor localization. In *Systems and Networks Communications, 2015. ICSNC'15. Eleventh International Conference on*, pages 25-29. IARIA, 2015.
- [7] Mário Melo and Gibeon Aquino. A taxonomy of technologies for fingerprint-based indoor localization. In *7o Simpósio Brasileiro de Computação Ubíqua e Pervasiva (SBCUP 2015)*, pages 111-120. SBC, 2015.
- [8] W. Najib, M. Klepal, and S.B. Wibowo. Mapume: Scalable middleware for location aware computing applications. In *Indoor Positioning and Indoor Navigation (IPIN), 2011 International Conference on*, pages 1-6, Sept 2011.
- [9] Anand Ranganathan, Jalal Al-Muhtadi, Shiva Chetan, Roy Campbell, and M Dennis Mickunas. Middlewhere: a middleware for location awareness in ubiquitous computing applications. In *Proceedings of the 5th ACM/IFIP/USENIX international conference on Middleware*, pages 397-416. Springer-Verlag New York, Inc., 2004.
- [10] M. Simoni, M.S. Jaakkola, L. Carrozzi, S. Baldacci, F. Di Pede, and G. Viegi. Indoor air pollution and respiratory health in the elderly. *European Respiratory Journal*, 21(40 suppl):15s-20s, 2003.

Improving risk identification process in project management

Allan de Godoi Contessoto, Laís Andreoli Sant' Ana, Rogéria Cristiane Gratão de Souza, Carlos Roberto Valêncio, Geraldo Francisco Donegá, Anderson Rici Amorim and Antonio Marcos Neves Esteca.

Department of Computer Science and Statistics, São Paulo State University, São José do Rio Preto, São Paulo Brazil.

{allan.contessoto, lais.andreoli, zafalon, anderson.rici, am.esteca}@sjrp.unesp.br, {rogeria, valencio}@ibilce.unesp.br

Abstract — Nowadays there are many information systems which aim the decision-making, automating the processes of project management. However, many times these systems limit the specific functions related to risk management processes and do not allow the control and effective treatment of uncertain events that may affect the project goals. Thus, this paper proposes the integrated use of the Delphi Technique and Risk Breakdown Structure – RBS in order to contribute to the identification of risks, considering the opportunities and threats in a project, to enhance the beneficial effects of the opportunities and avoid the harmful effects of threats. For the evaluation of the results, it was performed the improvement of a system called System to Aid Project Managing – SAPM, in order to support the automated execution of risk management with an emphasis on risk identification process, since this is a crucial stage for risk management in projects. As a result of this work, it is observed that the combined use of the Delphi Technique and RBS allows the risk identification process can be performed fast and safely through the anonymity of those involved and global visualization of the project.

Keywords - Risk Management, Delphi Technique, Risk Breakdown Structure, System to Aid Project Managing.

I. INTRODUCTION

The project management is becoming increasingly important in all sectors of the economy, since it provides an environment that allows users to manage the complexity of the projects, considering their restrictions and requirements [1]. A project can be understood as a temporary effort to create a product, service or exclusive result to satisfy the customer needs. Thus, project management is the application of knowledge, abilities, tools and techniques in the project processes, in order to achieve some goals [2].

The Project Management Body of Knowledge – PMBoK [2] describes an efficient management through the application of ten knowledge areas. Among them, there is the risk management applied to reduce the probability of occurrence of harmful events and creates answers to these events, besides increases the probability of occurrence of helpful events to the project and actions for achieving these events. A risk is defined as an event or an uncertain condition that, if it occurs, it will cause a positive or negative effect to, at least, one goal of the project. When a risk may cause a positive effect, it is named opportunity, however, when the risk may cause negative effect, it is named threat [2, 3].

In order to assist the decision-making of managers many computational tools have been developed [4, 5, 6]. However, few available tools have features that comply with the guidelines proposed by the PMBOK, primarily for the treatment of risk management, which results in difficulties for the correct application of this knowledge area.

In this scenario, the present work aims to promote the integrated use of support techniques to identify the risks efficiently and safely, contemplating the activities for the successful completion of the management of the opportunities and threats of a project. Thus, we adopted the Delphi Technique and Risk Breakdown Structure, which were incorporated into the System to Aid Project Managing - SAPM [6], aiming to expand its features for the correct evaluation of proposed risk identification process.

II. RISK MANAGEMENT

The risk management is proposed by PMBoK guide as a set of six processes, which interact among them and with other knowledge areas. Each process occurs at least once per life cycle of the project [2]. The risk identification process is crucial for the efficient execution of risk management in projects because the wrong identification of risks influences directly in the failure of a product or service, because it is difficult to manage something not well known [8]. Therefore, the project managers must promote actions to provide a complete and comprehensive identification of risks.

For the selection of risk identification techniques in this study were considered the main techniques described in the PMBoK guide because they are very accepted and disseminated among the professionals. As a result to improve the efficiency of risk identification process was proposed the use of Delphi Technique combined with Risk Breakdown Structure (RBS). Therefore, when a risk is identified by Delphi Technique and its characteristics are documented, it is essential to introduce of RBS to provide a project overview.

The Delphi Technique may be used to identify risks by consensus of specialists in order to obtain their opinion anonymously. Thus, the specialist, who is the mediator, uses a questionnaire to ask ideas about project risks. Then, the responses are summarized and redistributed anonymously to participants to assess each risk identified. After a few evaluation rounds, the consensus is obtained. Therefore, the Delphi Technique helps to reduce data partiality because it prevents personal influence on identified risks [2, 8].

The RBS was developed with the aim of interpret easily and directly the risk definition and assist managers to identify the risks in an organization. The RBS considers several risk sources to promote a vision over all parts of the project, and assist in the identification of recurrent themes and project areas with risk concentration. The RBS provides risk identification in a structured format and each lower level represents a more detailed definition of project risk sources [7].

In the literature there are many studies to provide the refinement and development of risk management [9, 10]. Due

to area advances, the RBS has been consolidated along the years as one of the essential techniques for correct application of risk management [7, 11]. However, the exclusive use of the RBS is not enough for the identification and characterization of risks, because it is a visualization technique. Thus, there is the need for the combination of RBS with some information gathering technique. Due to its anonymity characteristics, which enable change of opinions without personal influences in the results, the Delphi Technique is the best option for the combined use with the RBS. Therefore, it was found that the use of Delphi Technique with Risk Breakdown Structure allows managers to perform efficiently identification of threats and opportunities by a global view of the project.

III. METHODOLOGY

The methodological process adopted for the realization of the project comprises three stages. In the first stage, we have analyzed computational tools to support the risk management, in order to identify the main functionalities and show the strongest and the weakest points of each tool. The chosen tools are: Microsoft Project (MS-P) [12], NetProject (Net-P) [13], dotProject (dot-P) [14] and SAPM [6]. The theoretical support to analyze these tools was based on the processes from the risk management of PMBoK, in addition to relevant papers in risk management. As a result, we noticed the lack of computational mechanisms to help the identification of the project risks. Thus, it was possible to determine the functional requirements needed to enhance the SAPM in order to provide improvements to supply the shortcomings found on the analyzed tools.

In the second stage, it was developed the expansion of the SAPM, in order to consider the identified functional requirements to provide computational resources to support the identification of risks, through the development of an automated RBS and the execution of the Delphi Technique. For the development, we used open computational resources, as PHP, HTML and JavaScript languages, MySQL server to the database system management and Apache web server.

Lastly, in the third stage, the expansion of SAPM allowed performing an evaluation process which is divided into three distinct phases: the first one is to evaluate the efficiency of characterization of risks in threats and opportunities; the second aims the analysis of the ease of use of the new functions added to SAPM to risk identification; the third is to verify the general efficiency of the adopted methods to risk management.

The first evaluation phase was done by 16 professional from many areas that apply the risk management in their daily activities. This stage occurred to evaluate the significance of the functions responsible for the given treatment to the opportunities and to the threats, since in the correct risk management, threats and opportunities demand the same attention. The participants were trained to a correct utilization of the SAPM functions, as well as the focus of the conducted evaluation. The SAPM was available to use 15 days long. Then, the participants answered an evaluation form, scoring from 0 to 10 each evaluated function. Moreover, the participants could contribute with comments, emphasizing the

strongest and the weakest points identified, and the practicality of the functions as well.

The second evaluation phase was done by nine professionals divided into two groups, in order to use the mechanisms of risk identification in different contexts, resulting, thus, in two case studies. Each group received a practical training about the functions of SAPM, but more specifically about the functions related to the RBS and Delphi techniques. Thus, after the training, the two case studies were monitored by a period of 15 days, which the participants used the Delphi Technique and RBS to simulate the identification of risk on a dummy project. After this period, the participants answered the evaluation questions about the ease of use integrated of the Delphi and RBS techniques.

Finally, the third evaluation phase was done by 30 professionals that received training about all functions of the SAPM related to the correct application of risk management. In this evaluation stage, the participants used the SAPM for a period of 15 days, applying the risk management to their daily activities. Thus, as the previous stages, the participants answered an evaluation form with questions related to the use of the whole system. Therefore, in the third stage, it was analyzed all the functions of the risk management, considering the support that the set of functions provides to the correct applying of risk management, as well as whether the execution of these functions makes more agile and comprehensive the application of this knowledge area.

IV. RESULTS

This section brings the results obtained with the methodological process adopted as well as an analysis of them. Table 1 shows the results of the analysis performed for each tool, considering two aspects: the first one is related to the use or not of functions that represent crucial resources to an efficient risk management and the second one refers to the percentage of support that each tool offers considering the risk management processes described in the PMBoK guide.

Table 1 - Analysis of computational tools

	MS-P	Net-P	dot-P	SAPM
Functions				
Risk identification technique	No	No	No	No
RBS crating graphically	No	No	No	No
Documentation	Yes	Yes	Yes	Yes
Historical basis	No	Yes	No	No
Processes (%)				
To plan risk management	0	0	0	0
Risk identification	0	40	80	40
Qualitative risk analysis	0	0	80	0
To plan risk responses	0	20	60	0
To control plan	0	0	40	0
To plan risk management	0	0	0	0

With the results of this stage of empirical study, it is possible to notice that analyzed tools treat superficially the risk management, when considered the good practices proposed by PMBoK guide. Another constraint observed in

the analysis is the complete lack of computational resources to give efficient support to the characterization of risks during their identification, distinguishing threats and opportunities. For the correct application of risk management is essential that the treatment given to threats and opportunities is the same. Considering the risk is frequently associated to negative aspects in projects, i.e., the threats, the treatment of opportunities is compromised, reducing the effectiveness of the application of risk management in projects. Throughout the analyzes presented, a set of functional requirements was established to overcome the observed shortcomings.

The implementation had the aim to offer functions concerning risk management in the SAPM through the identified requirements. From the inclusion of an opportunity or a threat, where the characteristics are described, the RBS is automatically built. The Figure 1 shows the initial interface of RBS in list format. It is important to mention that the insertion of a threat or an opportunity occurs in the same form, which highlights the same approach for both.

During the first phase of the evaluation process we were able to confirm the relevance of the functions implemented in SAPM to approach opportunities and threats in the same way. In the histograms presented in the Figure 2 we can verify the evaluation of each criterion analyzed. Through the grades in the histograms it is possible to notice that the participants certified that the functions provided an efficient approach for opportunities and threats, simplifying the application of risk management in a project.

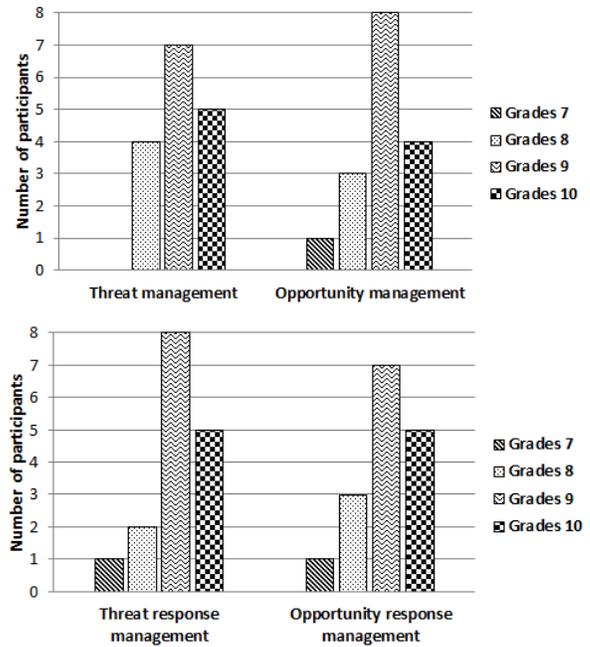


Figure 2 – Evaluation of treatment of opportunities and threats

The second phase of the evaluation process was performed through two case studies, i.e., Delphi Technique and RBS, to verify the efficiency and agility of approaches to identify the risks. In each case study has occurred an appointment of a mediator and the criteria for choice were education level and experience in risk management. Initially, in both case studies, each participant had the access to information related to the project and then they could develop a list of risks. In the following, the mediator analyzed all the identified risks to refine and eliminate potential redundancies, which resulted in a list of risk of mediator. After the phase of mediator's analysis, all the participants had the access to the list of risk proposed by mediator to add new risks and insert some comments. All process was performed during two complete rounds, because the mediators deemed that consensus had already been established. It is important to note that in huge and complex projects many iterations can occur to reach the consensus.

We observed in both case studies that were identified more threats than opportunities, being nine in the first case and five in the second one, while the opportunities were four and three respectively. This fact shows clearly the difficult of the participants to identify opportunities which can be explained by the negative connotations associated with term risk. Generally, risk gives an idea of something must be minimized, avoided, or that is related with danger or fault. Thus, it is possible to emerge that the inclusion of mechanisms to support the treatment of opportunities together with the treatment of threats are a good choice, proving a more comprehensive risk management.

Finally, with the results of the third phase is possible to observe the efficiency and the ease of use of SAPM's expansion. The Figure 3 shows the obtained results of this phase of the evaluation process.

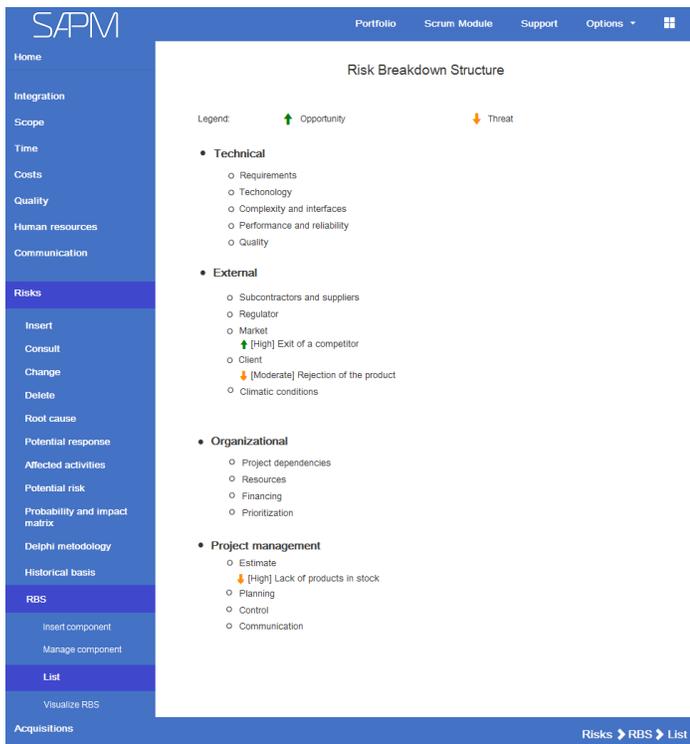


Figure 1 – RBS in a list format

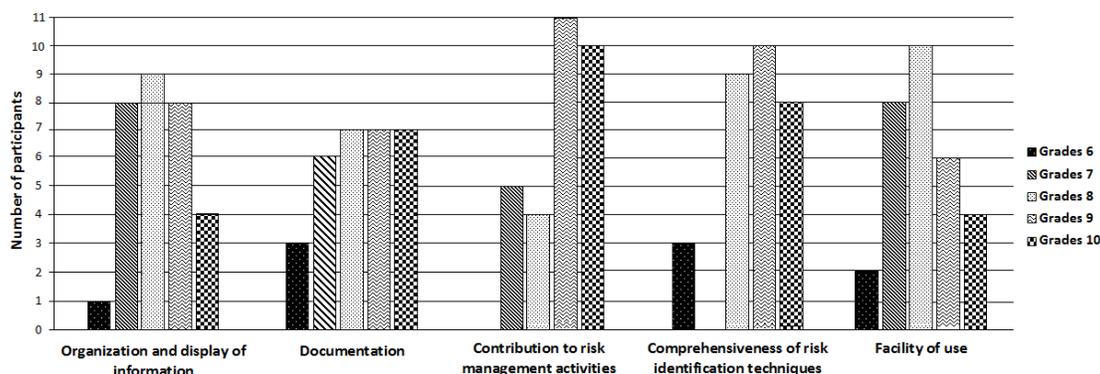


Figure 3 - General evaluation of SAPM's expansion

V. CONCLUSION

This paper presented mechanisms to support an efficient risk management in projects with the focus on the risk identification process. Thus, the control of project opportunities and threats offered can contribute to managers make decisions based on recent and reliable information.

The relevance of combined use of Delphi Technique with RBS was attested through the functionalities embedded in the SAPM which were evaluated by a three-step process. Such process reinforce the importance of functions implemented to perform the correct risk management following best practices proposed by PMBoK guide. Thus, it is possible to infer that the project risks can be properly identified, answered and supervised through the combined use of Delphi Technique and RBS.

The use of techniques to assist managers during risk identification is very important since faults in the implementation process compromises entire performance of risks management, which can cause difficulties for completion of the project objectives. The automated creation of RBS allows project managers visualizing the points that have the highest concentration of uncertainties during entire project life cycle. As perspective for future works, we intend to developed methods to assist managers for planning responses to the risks, as well as automated methods which indicates the probability of occurrence and impact. Moreover, should be set support resources to encourage the use of Risk Management Maturity Model [15] by organizations, through the definition of a method that can support the progress between the maturity levels set by the model. Finally, we are able to conclude the present study contributed to effective risk identification in a project with automated support through the expansion of SAPM, using Delphi Technique and RBS together.

VI. REFERENCES

[1] R. S. Pressman, "Software Engineering: A Practitioner's Approach", 7th edition, 2010.
 [2] PMBoK, "A Guide to the Project Management Body of Knowledge", 5th edition, Project Management Institute – PMI, 2013.
 [3] Gregoriades, A., Lesta, V. P., and Petrides, P., "Project Risk Management Using Event Calculus", In Proceedings of the 23th International Conference on Software Engineering and Knowledge Engineering (SEKE), p. 335-338, Miami, USA, 2011.

[4] Fontoura, L. M., and Price, R. T., "Systematic Approach to Risk Management in Software Projects through Process Tailoring" In Proceeding of the 20th International Conference on Software Engineering and Knowledge Engineering (SEKE), p. 179-184, São Francisco, USA, 2008.
 [5] Tosun, A., Bener, A., and Kocaguneli, E., "BITS: Issue Tracking and Project Management Tool in Healthcare Software Development", In Proceedings of the 21th International Conference on Software Engineering and Knowledge Engineering (SEKE), p. 526-529, Boston, USA, 2009.
 [6] R. C. G. Souza, A. M. N. Esteca, A. B. Santos, C. R. Valêncio, and M. T. Honda, "Web System to Aid Project Management", In Proceedings of the 23th International Conference on Software Engineering and Knowledge Engineering (SEKE), p. 325-330, Miami, USA, 2011.
 [7] D. Hillson, "Using a Risk Breakdown Structure in project management", *Journal of Facilities Management*, v.2, p. 85-97, 2003.
 [8] J. L. Worrell, P. M. Di Gangi, A. A. Bush, "Exploring the use of the Delphi method in accounting information systems research", *International Journal of Accounting Information Systems*, v. 14, n. 3, p. 193-208, 2013.
 [9] M. B. Junkes, A. P. Tereso, S. P. Afonso, "The importance of risk assessment in the context of investment project management: A case study", *Computer Science*, v. 64, p. 902 - 910, 2015.
 [10] S. Alhawari, L. Karadsheh, A. N. Talet, E. Mansour, "Knowledge - based risk management framework for information technology project", *International Journal of Information Management*, v. 32, n. 1, p. 50 - 65, 2012.
 [11] D. Hillson, "Using a Risk Breakdown Structure in project management", *Journal of Facilities Management*, v. 2, p.85 - 97, 2003.
 [12] Microsoft Project, <https://products.office.com/en-us/project/project-and-portfolio-management-software>.
 [13] NetProject, <http://www.netproject.com.br/>.
 [14] dotProject, <http://www.dotproject.net/>.
 [15] A. Serpell, X. Ferrada, L. Rubio and S. Arauzo, "Evaluating risk management practices in construction organizations", In *Procedia-Social and Behavioral Sciences*, v. 194, p. 201-210, 2015.

On Criteria to Choose a Content Management System: A Technology Acceptance Model Approach

Jislane S. S. Menezes¹, Danilo G. A. Ramos¹ and Michel S. Soares¹

¹Department of Computing, Federal University of Sergipe, São Cristóvão, Brazil
{jislanesds@gmail.com, dgaramos@gmail.com, mics.soares}@gmail.com

Abstract

Efficiently choosing software tools is a complex process, as it must consider many variables so that the best software which fulfills the requirements list can be selected. Choosing the correct Content Management System (CMS) for an organization is no different. Many different variables have to be taken into account, and large amount of resources have to be applied. However, mostly the process of choice is based on weak considerations. This paper uses the Technology Acceptance Model (TAM) in order to verify the influence of the evaluation criteria in the process of selecting a CMS. Through a descriptive statistical analysis and the Spearman's correlation coefficient, relations between the three concepts proposed by the TAM model, Perceived Usefulness (PU), Perceived Ease of Use (PEOU) and Perceived Usage (PUE), were evaluated. Data were submitted to non parametric methods and the results confirm two of the three formulated hypothesis: Perceived Ease of Use positively influences Perceived Usage of a CMS (H2) and Perceived Ease of Use positively influences Perceived Usefulness (H3). A case study performed on an IT government company regarding five instances of CMS is proposed in this paper.

Index Terms—Technology Acceptance Model, Content Management Systems, Software Acquisition

I. Introduction

Information Systems are considered critical success factors for most companies [1]. It is well-known that failures can lead even well-established organizations to

bankruptcy [2]. With the adoption of information systems by companies and the advent of Internet, organizations are capable of sharing information using web sites and working on projects remotely, connecting their collaborators from around the world. However, organizations have faced many issues for developing and maintaining their web based applications. Their solution is either hiring a technical team to develop and manage the web systems, or hiring a company to perform these tasks. Both solutions have shown to be problematic for reasons such as high costs, slow pace of change, and safety issues. Therefore, Content Management Systems (CMSs) appeared in the mid of the 1990's to facilitate development and management of organizations' web sites [3], as well as managing corporate data and content of organizations on the web.

Selecting the most useful Information System in an efficient way, when too many alternatives do exist, and when multiple criteria are known, is both hard and crucial [4]. Choosing the correct CMS for an organization is no different. Among so many CMS solutions, it is hard for a company to choose the right one after considering the company characteristics' and so many variables, such as cost, ease of use, stability, user comfort, among others, so that the best software which fulfills all the requirements can be selected to the client. In addition, time-pressure creates discomfort and limits the amount of time spent examining available information for decision making [5]. Software acquisition is considered a risky and complex activity [6]. According to results published in [7], risks occurred more frequently in software acquisition than in software development. Decision has also to consider "good enough" solutions, because the high number of criteria and most often the shortage of time makes it difficult to choose the best solution.

This paper creates a theoretical foundation for the development of a model to choose a CMS in order to meet

customer needs. The model is based on the Technology Acceptance Model - TAM. TAM was initially proposed to evaluate information systems tools [8] [9], but it has also been used in other contexts, such as to shape user attitude toward Mobile Cloud Services [10], to propose a model which explains and predicts the use of software measures [11], among other studies. In our case, we applied TAM to discover most interesting criteria to evaluate CMSs for users. A case study performed on an IT government company regarding five instances of CMS is proposed in this paper.

II. Background on CMS and TAM

A. CMS

A CMS [12] is a software used to develop and maintain websites in an easier way than using a programming language and other software development tools. By using a CMS, a user can create a website, update it, manage the content, post new content without changing the whole code of the page, or having to create new pages from scratch. Everything is managed by the CMS, which may include useful tools such as text editors, multimedia managing, social network integration, layout editors, blog management systems, users management systems and a wide variety of features that may vary from software to software.

There are many available CMSs. A simple search in a common search engine will find more than one hundred instances of CMSs, with many different features and characteristics. CMSs are an alternative to supply the demand of development and maintenance of websites without previous technical knowledge. As the years passed by, CMSs evolved into robust platforms of development that provide a high number of aggregate features, such as text editors, management tools, document management, authorization and workflow, customization and multi lingual publishing. They can be bundled or stand-alone applications and their main goal is to create, manage, deploy and store content on web pages, which can include multimedia files and applications that interact with the user.

B. TAM

Technology Acceptance Model - TAM [8] is a model created specifically for the study of the usage behavior of computer technologies. TAM model suggests that when users are presented with a new technology, a number of factors influence their decision about how and when they will use it. Originally, TAM incorporates two main variables, Perceived Ease of Use - PEOU, and Perceived Usefulness - PU. A third variable is considered in this research: Perceived Usage - PUE [9].

Perceived Usefulness - PU variable can be defined as “how the person believes that using a certain technology will improve its performance on a certain task” [8]. This is important because given the right idea of how users think the software will help them to achieve their goals on their tasks at work.

Perceived Ease of Use - PEOU variable can be defined as “how the person believes that learning to use a certain technology is without effort” [8]. It is important to know if there will be any resistance from the users to the new technology. Sometimes softwares are very difficult to use, frustrating the users and making their performance to finish their tasks worse than without the software.

Perceived Usage - PUE is the variable that seeks to point out the features that are really used in the software. This variable finds which are the features that have most important use in the tool. Given the right importance to the features, we can see how the software will be used, which features will be used, and the ones not considered by most users.

III. Methodology

A. Research Model Proposed

Since the goal of this work is to understand what criteria influence the attitude of users on the interaction with a CMS, a set of criteria has been grouped in TAM’s central variables.

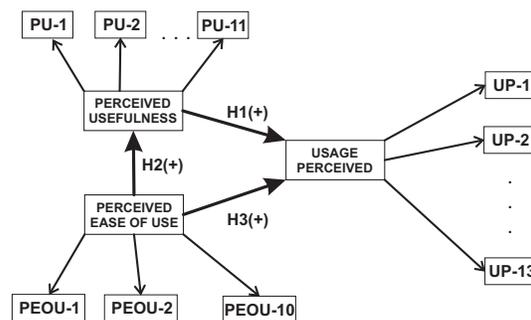


Fig. 1. Research model (adapted from Adams (1992) [9])

Figure 1 depicts the proposed research model. For the Perceived Usefulness variable, 11 criteria of measurement were set for basic maintenance of a content editor. For the variable Perceived Ease of Use, 10 evaluation criteria were set as learnability, ease of use on the interface, installation and update. For the Perceived Usage variable, 13 criteria were proposed that assess additional features plugins that tend to facilitate frequent interaction between software and user. Given the above model and the need to

understand how these variables are correlated, we present the following research hypotheses:

H1. Perceived Usefulness positively influences the Perceived Usage to a CMS.

H2. Perceived Ease of Use positively influences Perceived Usage to a CMS.

H3. Perceived Ease of Use positively influences Perceived Usefulness to a CMS.

B. Instrument Development and Validation

An online questionnaire consisting of items describing the study variables is used as data collection instrument. Design of the questionnaire was informed by literature review, the statement of problems, and research problems. A 5-point Likert scale was proposed to measure perceived attitudes of the employees by providing a range of responses to each statement. The scale ranged from (1) strongly disagree, (2) disagree, (3) neutral, (4) agree, and (5) strongly agree.

The questionnaire, distributed via email to employees of State government departments that interact with some instance of CMS, was designed after interview with experts from an IT Company, a government agency that centralizes development of web portals for all government departments. Selection of criteria noted the items considered important by experts and two user groups were selected: IT Department and Communication Department. Hereby, 20 questionnaires were completed and considered valid for statistical analysis. 34 questions were proposed in the survey, relating to three studied variables, two demographic questions and two questions about the degree of interaction with CMS tools. In the questionnaire, constituent definitions of each variable were described. Presented variables are: Perceived Use (PU - 11), Perceived Easy of Use (PEOU - 10), and Perceived Usage (PUE - 13).

This study employed quantitative methods for analyzing data. Data collected was subjected to a descriptive and inferential analysis involving mean, standard deviation and frequency count. Used technique is the Spearman correlation, since the direction of the relationship was specified in the hypothesis and variables were not normal.

IV. Results

A. Responses to TAM Questionnaire

By analyzing averages presented in Tables I, II and III, it turns out that participants of the poll have shown an elevated agreement in relation to most of the items from the survey. This mean they think the applications are very easy to use, are useful and that they used them very often. Besides, from data we can conclude it is common that more

than 80% of users have a positive opinion, (4 or higher) for most of the questionnaire. The positive assessment of respondents showed a 94.6% participation for PU, a 86.5% for PEOU and a 75.9% for PUE. With these results, it is noted that the technology is well received, they hope for software that is easy to use and make their work performance better with basic and additional features. Items that showed a rejection from the poll were PUE9 and PEOU8. Perceived Usefulness variable items showed the higher levels of agreement.

B. CMS Users' Perceived Ease of Use, Usefulness and Use Relations

In order to check the relations between the CMS users' perceived ease of use, usefulness and use, the Spearman correlation presented the existence of a significant and positive correlation (Sig < 0.05) to relations between PEOU and PU, and between PEOU and PUE. Table IV shows this information more clearly.

Variable Rho values to H2 and H3 hypotheses were very close to H1, which shows a moderate and strong association between variables Perceived Ease of Use and Perceived Usage, and between Perceived Usefulness and Perceived Ease of Use, respectively. Hypothesis H1 has not been confirmed, since it did not achieve less than 0.05 [13] of significance value and Rho's coefficient shows a low correlation. Therefore, even tough users understand that the CMS tool will improve their performance on work activities (PU), this benefit does not relate to the perception of easiness to use the CMS without effort (PEOU). Differently from the other two hypothesis which highlight the easiness of using (PEOU) as an important aspect of a CMS acceptance, as it influences the other variables of this proposal.

Result of hypothesis H3 is corroborated by the study of [14] which confirms the positive influence between Perceived Ease of Use and Perceived Usefulness in more than 75% of assessed works. No works evaluating the Perceived Usage variable proposed by Adams [9] in relation to other variables were found. With this, hypothesis H2 and H3 can be supported.

V. Case Study

Five popular Open Source CMS's were chosen to be part of the evaluation process. The evaluation process took ten weeks long, with approximately two weeks per CMS, since the TAM Method requires a certain use and knowledge of the software being analyzed.

For the analysis results, the CMS's are named as CMS1, CMS2, CMS3, CMS4 and CMS5, being CMS1 the first CMS to be evaluated, CMS2 the second one and so on.

TABLE I. Descriptive statistics PU (N = 20).

Perceived Usefulness - statements 1 to 11	1	2	3	4	5	m	s	pos
1 - The CMS should allow webpage edition	0	0	1	9	10	4.45	0.61	19
2 - The CMS should allow update of content that was published previously	0	1	0	8	11	4.45	0.76	20
3 - The CMS should allow indexing of stored documents	0	0	0	10	10	4.5	0.51	20
4 - The CMS should allow organization of content by subject	0	0	0	11	9	4.45	0.51	20
5 - The CMS should allow organization of content via menu	0	0	0	10	10	4.5	0.51	20
6 - The CMS should provide interactive content containing videos	0	0	2	8	10	4.4	0.68	18
7 - The CMS should provide interactive content containing audio	0	0	3	8	9	4.3	0.73	17
8 - The CMS should provide a news area	0	0	0	6	14	4.7	0.47	20
9 - The CMS should provide layout edition without changing directly source code of the page	0	1	4	8	7	4.05	0.89	16
10 - The CMS should provide a text edition tool	0	0	0	10	10	4.5	0.51	20
11 - The CMS should be compatible with most web browsers.	0	0	0	5	15	4.75	0.44	20

TABLE II. Descriptive statistics PEOU (N = 20).

Perceived ease of use of CMS - statements 1 to 10	1	2	3	4	5	m	s	pos
1 - The CMS should be easy to learn	0	0	1	8	11	4.5	0.61	19
2 - The CMS should be easy to install	0	0	2	9	9	4.35	0.67	18
3 - The CMS should be easy to install in various operational systems	0	0	3	9	8	4.25	0.72	17
4 - The CMS should be easy to update	0	0	0	6	14	4.7	0.47	20
5 - For me, using a CMS is often frustrating	0	8	6	4	2	3	1.02	14
6 - The CMS should be flexible to interact	0	1	0	14	5	4.15	0.67	20
7 - The CMS should facilitate memory on how to make tasks	0	0	3	12	5	4.1	0.64	17
8 - The CMS should have a self explanatory interface	0	0	3	10	7	4.2	0.7	17
9 - The CMS should facilitate insertion of images on a text	0	0	0	10	10	4.5	0.51	20
10 - The CMS should facilitate text diagramming	0	0	0	8	12	4.6	0.5	20

TABLE III. Descriptive statistics PUE (N = 20).

Perceived Usage - statements 1 to 13	1	2	3	4	5	m	s	pos
1 - The use of file sharing methods on the CMS increases my performance at work	0	0	6	8	6	4	0.8	14
2 - The notification by e-mail in interactions in new posts increases my performance at work	0	2	5	9	4	3.75	0.91	15
3 - The comment approval feature increases my performance at work	0	2	6	8	4	3.7	0.92	14
4 - The reply comments feature increases my performance at work	0	0	6	9	5	3.95	0.76	14
5 - The classification of documents stored on the system increases my performance at work	0	0	4	11	5	4.05	0.69	16
6 - The use of content search tools combining categories and keywords makes my work easy	0	0	1	10	9	4.4	0.6	19
7 - The discussion forums feature increases my performance at work	0	1	8	9	2	3.6	0.75	12
8 - The poll creation feature increases my performance at work	0	0	5	9	6	4.05	0.76	15
9 - The blog creation feature increases my performance at work	1	0	11	5	3	3.45	0.95	9
10 - The remote access for content edition increases my performance at work	1	0	1	10	8	4.2	0.95	19
11 - The RSS feeds feature makes my work easier.	0	0	5	11	4	3.95	0.69	15
12 - The integration with social networks (Facebook, Twitter) increases my performance at work	0	1	1	7	11	4.4	0.82	19
13 - The link incorporated to text creation makes my work easier	0	0	1	7	12	4.55	0.61	19

TABLE IV. Correlation between hypothesis and Spearman coefficient

Hypothesis	Rho Coefficient	Sig.
H1 - Perceived usefulness positively influences perceived usage to a CMS	.389	.090
H2 - Perceived ease-of-use positively influences perceived usage to a CMS.	.468	.038
H3 - Perceived ease-of-use positively influences perceived usefulness to a CMS.	.681	.001

Each CMS was analyzed by two evaluators. The two evaluators analyzed the CMS's independently and evaluated them via an online form. When the analysis was finished, each evaluator filled the form based on their own thoughts about the CMS. The form had the questions previously shown and explained in the TAM section of this paper, and the answers were in a 3 point scale.

In order to create the test environment on PHP based

CMS's, Wampserver 2.5 was used as ambient. Wampserver is an application which installs a web development ambient with Apache 2.4.9, PHP 5.5.12 and MySQL 5.6.17, it also comes with PHPMyAdmin as a tool to facilitate database management. On the Python based CMS the tools were a Vagrant kit with a VirtualBox virtual machine and the Putty SSH to simulate a web server to run the CMS. Tables V, VI and VII represents the results of the CMSs evaluation using

TABLE V. Resulting Table PU

PU	CMS1	CMS2	CMS3	CMS4	CMS5	Weight
1	1	1	0.5	1	1	4.45
2	1	0.75	1	1	1	4.45
3	1	1	0.75	0.75	1	4.5
4	1	1	0.75	1	1	4.45
5	1	1	1	1	1	4.5
6	1	1	1	0.75	1	4.4
7	0.75	1	0.75	0.75	0.75	4.3
8	1	0.5	0.5	1	1	4.7
9	1	0.25	0	0.75	0.25	4.5
10	1	1	0.5	1	1	4.05
11	1	1	1	1	1	4.5

TABLE VI. Resulting Table PEOU

PEOU	CMS1	CMS2	CMS3	CMS4	CMS5	Weight
1	1	0.25	0	0.75	1	4.5
2	1	0.75	0.5	0.25	0.25	4.35
3	1	0.75	0.75	0.5	0.25	4.25
4	0.75	0.5	0.75	0.5	0.5	4.7
5	1	0.5	0	0.5	1	3
6	1	0.5	0.25	1	0.75	4.15
7	1	0	0	1	1	4.1
8	1	0.25	0.25	1		4.2
9	1	1	1	1	1	4.5
10	1	0.5	0.5	1	1	4.6

TABLE VII. Resulting Table PUE

PUE	CMS1	CMS2	CMS3	CMS4	CMS5	Weight
1	1	1	0.75	1	1	4
2	0.5	0.25	0.25	0.25	0.5	3.75
3	1	0	0.5	0.5	1	3.7
4	1	0.25	1	1	1	3.95
5	0.75	1	0.75	0.75	1	4.05
6	0.75	0.75	0.5	1	0.75	4.4
7	0.5	0.5	1	0.5	0.5	3.6
8	0.5	0.5	0.25	0.5	0.25	4.05
9	1	0.5	0.5	0.75	0.5	3.45
10	0.5	0.5	0.5	0.75	0.75	4.2
11	1	1	1	1	1	3.95
12	1	0.75	0.5	0.5	0.75	4.4
13	1	1	1	1	1	4.55

the validated TAM criteria by the interviewed subjects on the Section Results of this paper. Each mean will be a weight in order to evaluate the CMSs.

Each item was evaluated receiving value 0, 0.5 or 1. As each CMS was evaluated by two evaluators, the final value was the mean between the evaluations. These tables serve as a guideline to help stakeholders to define which CMS is better considering their purposes and requirements. In order to calculate the final scores for each CMS, each evaluated value was multiplied by the mean found by TAM-based questionnaire, presented in Section Results of this paper. Fig. 2 depicts the final scores for each CMS, for each TAM variable.

Following the graphic of Perceived Usefulness (PU), the evaluated softwares answer above 70% of the items

featured in Table 3, highlighting the CMS1 with 97.8% of agreement in relation to the other CMSs. In relation to Perceived Ease of Use, CMS1 got 97.2% in the total evaluation. Hence, this CMS shows advantages in items such as Ease of Use and deployment, which decreases the learning curve. CMS3 did not get the best results, specially for the difficulty to find manuals and tutorials available for research, besides being non-intuitive in most operations. As for Perceived Usage, the CMSs attended 81%, 63% and 65.5% respectively. This result shows that the softwares attended a reasonable quantity of features considered important by the interviewees, as for instance, document classification, the incorporation of hyperlinks in the text and social networks integration.

VI. Threats to Validity

This study has endeavored to establish a research model for the successful evaluation of CMS systems at an IT government company. However, the study has some limitations that can be addressed in future research. The study had 20 people surveyed, which is a relatively low number for a reliable estimate. However, the survey was answered by subjects working at various state organizations and departments. The method presented in this paper can be replicated with more respondents by other researchers. In terms of scope, this study is confined to the State government departments. Future research could include more public and private institutions that are using CMS.

The response categories in Likert scale has rank order, but the intervals between values cannot be presumed equal [15]. As a result, Likert scales fall within the ordinal level of measurement. When treated as ordinal data, Likert responses can be analyzed using non-parametric tests, such as the Mann-Whitney test, the Wilcoxon signed rank test, and the Kruskal-Wallis test [16]. Non-parametric statistical methods were used to analyze the findings because the subjects involved in the study were few and not chosen randomly from a large population. Objective here was to find representative power of results, and not necessarily statistical significance. It is worthwhile to note that non-parametric techniques do not solve the problem of potential dependency between the answers of participants.

VII. Conclusion

The process of choosing a software application is a critical factor observed by companies. Large amount of time and financial resources have to be applied for software acquisition. Mostly, the process of choice is based in weak considerations. Considering that, this research aimed to establish evaluations criteria based on TAM, which is an

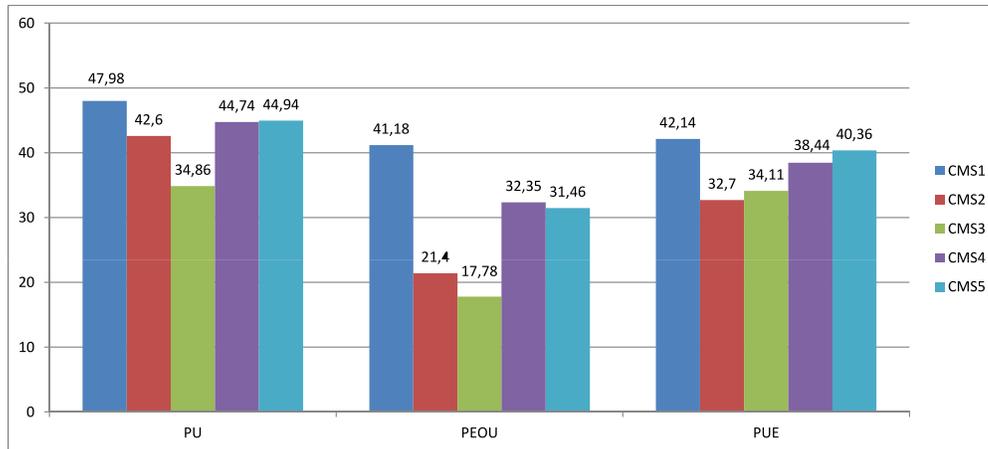


Fig. 2. Final results.

empirical evaluation method applied to comprehend use and behaviour from users of a certain technology.

The research had as purpose to verify the influence of CMS tools evaluation criteria on the relationship between acceptance and use of those systems. It can be seen in this research that users who interact with CMS systems are stimulated to use the product due to the Perceived Ease of Use, the Perceived Usefulness and the Perceived Usage. Whereas there were no rejections from the analyzed variables, it was also observed that the variable Perceived Usage is influenced by variable Perceived Ease of Use. These findings allow to identify what criteria should be considered during a successful evaluation process to choose a CMS tool. For future research, the relation between some variables and user profile can be analyzed.

Acknowledgment

The authors would like to thank research agency FAPITEC/SE (grant 9614), Governo do Estado de Sergipe, and NAP-CPD EMGETIS (<http://www.emgetis.se.gov.br>).

References

- [1] S. Petter, W. H. DeLone, and E. R. McLean, "Information Systems Success: The Quest for the Independent Variables," *J. of Management Information Systems*, vol. 29, no. 4, pp. 7–62, 2013.
- [2] Y. Dwivedi, D. Wastell, S. Laumer, H. Henriksen, M. Myers, D. Bunker, A. Elbanna, M. Ravishankar, and S. Srivastava, "Research on Information Systems Failures and Successes: Status Update and Future Directions," *Information Systems Frontiers*, pp. 1–15, 2014.
- [3] M. White, *The Content Management Handbook*, 1st ed. Facet Publishing, 2005.
- [4] F. Daneshgar, G. C. Low, and L. Worasinchai, "An Investigation of Build vs. Buy Decision for Software Acquisition by Small to Medium Enterprises," *Information and Software Technology*, vol. 55, no. 10, pp. 1741–1750, 2013.
- [5] D. Fehrenbacher and S. Smith, "Behavioural Affect and Cognitive Effects of Time-pressure and Justification Requirement in Software Acquisition: Evidence from an Eye-Tracking Experiment," in *20th Americas Conference on Information Systems*, 2014.
- [6] S. Harnisch, "Enterprise-level Packaged Software Acquisition: a Structured Literature Review Through the Lens of IT Governance," in *European Conf. on Information Systems*, April 2014.
- [7] D. Kusumo, M. Staples, L. Zhu, H. Zhang, and R. Jeffery, "Risks of Off-the-Shelf-Based Software Acquisition and Development: A Systematic Mapping Study and a Survey," in *16th Int. Conf. on Evaluation Assessment in Software Engineering*, May 2012, pp. 233–242.
- [8] F. D. Davis, "Perceived Usefulness, Perceived Ease of Use, and User Acceptance of Information Technology," *MIS Quarterly*, vol. 13, no. 3, pp. 319–339, 1989.
- [9] D. A. Adams, R. R. Nelson, and P. A. Todd, "Perceived Usefulness, Ease of Use, and Usage of Information Technology: a Replication," *MIS Quarterly*, vol. 16, no. 2, pp. 227–247, 1992.
- [10] E. Park and K. J. Kim, "An Integrated Adoption Model of Mobile Cloud Services: Exploration of Key Determinants and Extension of Technology Acceptance Model," *Telemat. Inf.*, vol. 31, no. 3, pp. 376–385, 2014.
- [11] L. G. Wallace and S. D. Sheetz, "The Adoption of Software Measures: A Technology Acceptance Model (TAM) Perspective," *Information Management*, vol. 51, no. 2, pp. 249–259, Mar. 2014.
- [12] S. Baxter and L. Vogt, "Content Management System," 2002, uS Patent 6,356,903. [Online]. Available: <https://www.google.com/patents/US6356903>
- [13] A. Field, *Discovering Statistics using SPSS*. Sage publications, 2009.
- [14] P. Legris, J. Ingham, and P. Collette, "Why do People Use Information Technology? A Critical Review of the Technology Acceptance Model," *Information & management*, vol. 40, no. 3, pp. 191–204, 2003.
- [15] S. Jamieson, "Likert Scales: How to (ab)use them," *Med. Educ.*, vol. 38, no. 12, pp. 1217–1218, 2004.
- [16] G. W. Corder and D. I. Foreman, *Nonparametric Statistics for Non-Statisticians: A Step-by-Step Approach*. Hoboken, NJ, USA: John Wiley & Sons, 2009.

Global Software development: An Approach to Design and Evaluate the Risk factors for Global Practitioners

Chamundeswari Arumugam
Department of Computer Science and Engineering
SSN College of Engineering
Tamil Nadu, India
chamundeswaria@ssn.edu.in

Baskaran Kaliamourthy
Managed Services, Engineer Manager
Ericsson Inc, Plano,
Texas, USA
baskaran@outlook.in

Abstract—In today's competitive environment organizations are increasingly moving towards global software development to utilize every possible benefits across geography. Practitioners from different part of the world are undertaking such assignments for multiple benefits like global exposure, new domain/arena to explore, better compensation, etc. In spite of the above benefits these global practitioners face multiple challenges due to cultural difference, communication, geographical dispersion.

This paper emphasizes the risk factors on global practitioners in an IT organization. Dependencies of the global practitioners are explored to identify the risk factors using four stages of dynamics model [30]. A Socio Technical Systems (STS) responsibility model is designed for global practitioners to exhibit the relationship between global practitioners within an organization structure and represent the risk of the global practitioners using a set of graphical notations. In addition this paper also proposes a MultiAgent Simulation Model (MASM) to evaluate the risk of global practitioners on gaining trust from project stakeholders. This work highlights the sequential and concurrent activities based on schedule and time for evaluating the risk.

Keywords - *global practitioners; socio technical system; multiagent simulation model; global system development*

I. INTRODUCTION

The rapid advancement in the Information Technology (IT) has widely spread the use of the Global Software Development (GSD) across the world. GSD is popular among the IT organization and a large number of IT employees are taking up global assignment, due to its benefits it offers, irrespective of the duration. Reduced development costs, leveraging time-zone effectiveness, cross site modularization of development work, access to large skilled labor pool, innovation and shared best practice, and close proximity to market and customer are some of the key benefits of GSD [6]. Many IT employees around the global are migrating from one region to another irrespective of the culture, language, security, food, etc. for GSD projects, and they are termed as global practitioners in this paper.

Many organizations have set up their branches in many cost economical different countries to outsource their projects. GSD differs from Distributed Software Development (DSD), as the development happens across national geographic boundaries. Development of projects across national geographic boundaries sets the cause of all challenges in GSD for global practitioners.

Global practitioners can be categorized as independent workers and employees of globally distributed companies. The challenges are different across these two categories. This paper focuses on the challenges of global practitioners in globally distributed companies. Global practitioners play a vital role in every phase of software development life cycle. This paper attempts to take a look at the risk associated with global practitioner in GSD projects, to gain trust of all stakeholders through conceptual model and Socio Technical System (STS) towards designing responsibility model to identify the risk for global practitioners and thereafter the possibility to apply MultiAgent Simulation Model (MASM) to evaluate them. The MASM can be used to evaluate the probability and impact of the risk by the global practitioners, to gain trust from stakeholders.

The organization of this paper is as follows. Section II provides an overview of related work in GSD risk and STS. Section III explains the research methodology in detail. Finally, Section IV concludes the paper and discusses limitations as well as future work.

II. RESEARCH METHODOLOGY

Global practitioners are selected by the organization based on the experience level, skill set, work division, domain knowledge, competency level, domain experience, time zone difference, cultural difference, process maturity diversity, critical functionality, commitment, etc. The selected global practitioners, has risk to gain trust from stakeholders in a GSD team.

The four stage dynamics model specified in Pressman [23] are considered as information resources to identify the risk factors. Forming, storming, norming, and performing are the four stages used as information resources to identify the risk

DOI reference number : 10.18293/SEKE2016-077

factors. During forming stage, an effective work culture should be set, ignoring conflicts and assigning commitment among the members. Setting up, work culture is a risk and it need to done to ensure commitment.

In storming stage, a well growth oriented foster relationship should be set, with close mind members in a project. Also the members should organize task and add value to the team. Setting up a foster relationship is risk and it is advisable, as each member can add value to his team. In norming stage, transparent process should be encouraged, to permit team members to take decision. Encouraging transparent process is risk but it enhances knowledge sharing.

TABLE I. IDENTIFICATION OF RISK FACTORS

Information resource	Risk factor	Dependent factors
forming	work culture	conflicts ⁻ commitment ⁺
storming	foster relationship	close minds ⁻ organize task ⁺ member value ⁺
norming	transparent process	knowledge sharing ⁺ team feedback ⁺
performing	Visibility	shared goals ⁺ optimal solution ⁺

In performing stage, more visibility needs to be created, as they work on shared goals to produce optimal solution. Creating a visibility is a risk, but it promotes to produce optimal solutions. The identified risk factors are represented in Table 1.

Conceptualization of a problem is the most important activity in the development of a system dynamics model [21]. Scholl in 1995 [22] implied that four mapping tools, causal loop diagrams, stock and flow diagrams, sector diagrams, and policy structure diagrams, are frequently combined in system dynamics projects. In this work, the identified risk factors are conceptualized using causal loop diagram. Vensim tool [24] is used to represent the conceptualization model using causal loop diagram. Causal loop diagram is represented in Figure 1. Using the identified risk factors, the STS responsibility model is derived and it is discussed below.

The concept of a “socio-technical system” (as part of “organizational development” research) was created in the 1960s by E. Trist and F. Emery (both of the Tavistock Institute for Social Research in London). This term “socio-technical system” basically considers the interaction of people in a social system or organization, with tools and technique, in a technical system. Baxter [2] specified six key characteristics of socio-technical systems. Cherns [5] stated four key principles of socio-technical systems.

Responsibility modeling [13] is a graphical modeling and analysis technique designed to help people record and analyze responsibilities within organizations, to explore the structure and dependability of socio-technical systems. Russell lock et al. [10] specified the key components of a responsibility model are

responsibilities, agents, and shared information. In this paper, responsibility modeling technique is used to represent the design of the identified risk factors. The key components identified for the responsibility model are information resource, risk factor, and multiagent. Here the multiagent is used to analyze and access the interactions among the various risk factors that happen between four information resources.

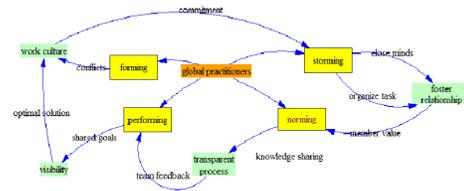


Figure 1. Conceptual model using causal loop diagram

The key component uses five symbols to represent relationship. The five symbols actually used for representation are curved rectangle, parallel line, greater and lesser sign, filled in circle, filled in diamond. Curved rectangle represents risk factor, parallel lines denote information resource, and greater and lesser sign denote multiagent. The other two symbols, filled in circle and hollow diamond are used to represent the relationship between risk factor, information resource, and multiagent. Figure. 2 represent the STS responsibility model for three key components with their corresponding relationship.

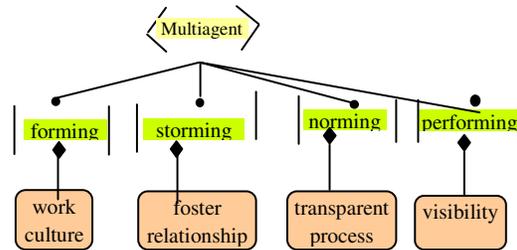


Figure 1. Responsibility model

A. Risk Assessment Model (RAM)

Multi-Agent Simulation Model (MASM) can provide a way for stakeholders to experiment with the different configurations [7]. Agent based simulation was already applied in software development projects for simulation purpose to forecast the performance of individual team members in a software development team [1]. The goal of MASM in this proposed work is to evaluate the information resources dynamic changes for risk assessment.

MASM was practically applied in software industry in certain environment where the complete development may procure cost. Considering this factor, an attempt has been done in this paper to apply MABS for risk assessment. Here, MASM can be applied to assess the risk for the global practitioners to gain trust from stakeholders, considering the four information resources to simulate the dynamic data. A

complex relationship prevails between the risk factors, and dependent factors, onto which multiagents can be applied. Figure 3. represents the agents flow direction to collect the data from the identified risk factors for risk evaluation.

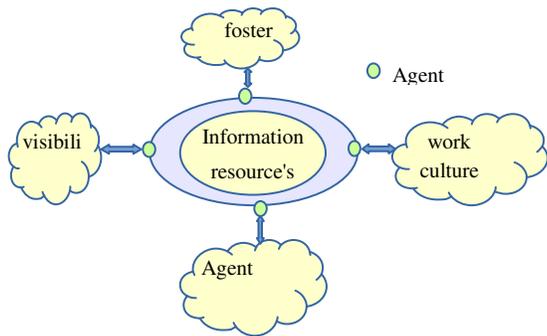


Figure 2. Flow of agent to collect the data from information resource

A single agent can perform one or more activities associated with information resources for risk assessment. An activity correspond a set of tasks to assess the risk. The simplest task represents a single evaluation in RAM. Each activity is attached with constraints. The constraints can be sequential order or time based event. The nesting of the events in sequential order and time event can be complex and this is referred as concurrent activities. Each agent is independent, and it has the responsibility to start an activity and also has privilege to interact with one or more agents. In multiagent concept, for interaction, the agents mutually agree and proceed. An agent has the right to accept or reject the interaction.

Figure. 4 show the proposed approach of using MASM for assessing the risk, a Risk Assessment Model (RAM). An agent associated with an information resource has many sequential and concurrent activities. An agent through execution control interacts either with sequential or concurrent activities of model, while data analysis collects and record the data for the various interaction between agents, for risk assessment.

III. LITERATURE SURVEY

Nils Brede Moe et al. [3] indicated that trust is a fundamental factor in determining the success or failure of GSD projects. Some of the identified key factors to be poor socialization and socio-cultural fit increased monitoring, inconsistency and disparities in work practices, reduction of and unpredictability in communication, lack of face-to-face meetings, language skills, conflict handling, and cognitive-based trust. The effect of lacking trust has implication for software managers, and practitioners involved in GSD. Christof Ebert et al. [8] identify and mitigate the GSE risk for development projects and product evolution.

Responsibility modeling [10, 12] is a graphical modeling technique designed to allow end users to model and explore many of the high level risks within their socio-technical systems without recourse to expensive risk management specialists. Indira Nurdiani et al. [11] conducted a systematic review on GSD literature and gathered challenges associated

with GSD projects as well as their mitigation strategies. The authors developed two static checklists for risk identification and mitigation. However, the author pointed out, they cannot be used as decision making tool to select the most appropriate mitigation for particular risk. Ansgar Lamersdorf et al. [9] assessed risks for task allocation alternatives based on project and site-specific characteristics and analyzed it with respect to possible project risks stemming from the work distribution.

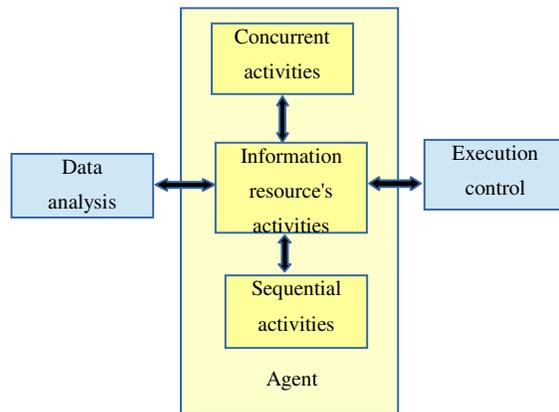


Figure 3. Risk assessment model using MABS

Verner et al. [15] has investigated GSD to discover the risk and mitigation advice for the organizations involved with GSD. The authors extracted 85 risks and 77 risk mitigation advice items and categorized them under four major headings: outsourcing rationale, software development, human resources, and project management. Muhammad Usman et al. [14] identified communication is a very important risk factor in GSD. Davy de Medeiros Baia [7] aims to create an integrated multi-agent-based simulation to support software project management.

Mohammed Alshammri [17] presents a research plan to develop combined agent-based and system dynamics models to simulate software development process focusing on the behaviour of team members in a software project environment. Kellner et al. [18] has clearly stated why, what, how to simulate. Wickenberg et al. [19] investigated the applicability of multiagent based simulation to simulate software development processes. David et al. [20] provided a set of general guidelines when to use simulation in software development process.

From the literature work, it is understood, identification and mitigation of risk based on GSD is important in GSD projects. However, in the present scenario, the global practitioners are the important contributor in GSD project. Thus, this paper proposes an approach to design and evaluate the risk factors, for a global practitioners in a GSD team to gain the trust of the stakeholders.

IV. CONCLUSION

This paper attempt to take a look at the risk associated with global practitioner in the GSD projects to gain trust from stakeholders. This paper proposes an approach to identify and

assess the risk factors, for global practitioners in GSD projects using four stage dynamics model [30]. A conceptual model that combines the dynamics for global practitioners in GSD projects, using the four stages, forming, norming, storming and performing, is attempted to identify the risk. Causal loop diagram mapping tool captures the dynamics and it is represented using the Vensim tool. Design using a socio technical responsibility modeling technique is represented to exhibit the relationship between global practitioners within an organization using a set of notations. Also, Risk Assessment Model (RAM) using MultiAgent Simulation Model (MASM) design is proposed to assess the risk factors. Assessing these risk factors is a measure for the global practitioners in the GSD projects.

ACKNOWLEDGMENT

We thank Dr. S. Sivakumar, Professor, SSN College of Engineering, Tamil Nadu, India for his continuous support and value added comments in drafting this paper.

REFERENCES

- [1] R. Agarwal, D. Umphress. "A Flexible Model for Simulation of Software Development Process," 48th Annual Southeast Regional Conference, 4 pages. 2010, New York, NY: ACM.
- [2] G. Baxter, I. Sommerville, Socio-technical systems: From design methods to systems engineering. Submitted to The journal of human-computer studies, 2008.
- [3] N.B. Moe, D. Smite, "Understanding a Lack of Trust in Global Software Teams: A Multiple-case Study," Software Process Improvement and Practice, vol 13, pp 217-231, 2008.
- [4] I.Bider, H.Otto,
- [5] A.B. Cherns, The principles of socio technical design, Human Relations, vol. 29 (1976), pp. 783–792.
- [6] E. Ó Conchúir, P. J. Ågerfalk, H. H. Olsson, and B. Fitzgerald, "Global software development: where are the benefits?", Communications of the ACM, vol. 52, pp. 127-131, 2009.
- [7] D. D. M. Baia, "An Integrated Multi-Agent-Based Simulation Approach to Support Software Project Management," 2015 IEEE/ACM 37th IEEE International Conference on Software Engineering, Doctoral Symposium, 2015.
- [8] C. Ebert, B. K. Murthy, N.N. Jha, "Managing Risks in Global Software Engineering: Principles and Practices," IEEE International Conference on Global Software Engineering, 2008, pp.131-140.
- [9] A. Lamersdorf, J. Münch, A. F. V. Torre, C.R. Sánchez, "A Risk-driven Model for Work Allocation in Global Software Development Projects," 2011 Sixth IEEE International Conference on Global Software Engineering, pp. 15-24.
- [10] R.Lock, I.Sommerville , "Modelling and Analysis of Socio-Technical System of Systems", 15th IEEE International Conference on Engineering of Complex Computer Systems, 2010, pp. 224-232.
- [11] I. Nurdiani , R. Jabangwe, D.Smite, D.Damian, "Risk Identification and Risk Mitigation Instruments for Global Software Development: Systematic Review and Survey Results," Sixth IEEE International Conference on Global Software Engineering Workshops, pp. 36-41, 2011.
- [12] R. Lock, I. Sommerville, Socio Technical Systems Engineering Handbook.
- [13] D. Šmite and J. Borzovs, "Managing Uncertainty in Globally Distributed Software Development Projects," University of Latvia, Computer Science and Information Technologies, vol. 733, pp. 9-23, 2008.
- [14] M. Usman, F. Azam, N. Hashmi, Analysing and reducing risk factor in 3-C's model communication phase used in global software development, International Conference on Information Science and Applications (ICISA), pp. 1–4 (2014)
- [15] J. M. Verner, O.P. Brereton, B.A. Kitchenham, M. Turner, M. Niazi, Risks and risk mitigation in global software development: A tertiary study Information and Software Technology, 56 (2014), pp. 54–78.
- [16] A. Zeid and R. El-Bahey , "Establishing a global software development course: A cultural perspective" , Frontiers in Education Conference, pp.1695 -1701, 2013.
- [17] M. Alshammri, Simulation Modelling of Human Aspects in Software Project Environment, ASWEC ' 15 Vol. II, September 28 - October 01, 2015, Adelaide, SA, Australia.
- [18] M. I. Kellner, Raymond J. Madachy, and David M. Raffo, "Software Process Modeling and Simulation: Why, What, How," Journal of Systems and Software, Vol. 46, No. 2/3 (15 April 1999).
- [19] T. Wickenberg, and P. Davidsson. 2003. "On Multi Agent Based Simulation of Software Development Processes." In Multi-Agent-Based Simulation II, Edited by J. Simão Sichman, F. Bousquet, and P. Davidsson, vol. 2581, 72–77, Berlin: Springer Berlin-Heidelberg.
- [20] D. Joslin, and W. Poole, Agent-Based Simulation For Software Project Planning, Proceedings of the 2005 Winter Simulation Conference.
- [21] R. Eberlein, V. Diker, R. Langer, J. Rowe (Eds.), Proceedings of the 21st International Conference of the System Dynamics Society. July 20–24, 2003 New York City, NY (2003).
- [22] Scholl, G. J. (1995). "Benchmarking the system dynamics community: research results." System Dynamics Review 11(2): 139 to 155.
- [23] Pressman, R. Software Engineering: A Practitioner's Approach. McGraw-Hill, (2005)
- [24] www.vensim.com

Problem-Aware Traceability in Goal-Oriented Requirements Engineering

Grace Park, Lawrence Chung
University of Texas at Dallas
{exp130530, chung}@utdallas.edu

Jang-Eui Hong
Chungbuk National University
jehong@chungbuk.ac.kr

José Luis Garrido, Manuel Noguera
University of Granada
{jgarrido, mnoguera}@ugr.es

Abstract— Requirements traceability helps to ensure that a requirements specification is aligned with the intended stakeholders' needs. Such alignment should involve the consideration of why such needs arise, in terms of what problems the stakeholders are faced with, and what kinds of software system may help alleviate or eliminate the problems. However, little work can be found on requirements traceability that explicitly considers the problems. In this paper, we propose a problem-aware framework for establishing requirements traceability, in the context of goal-oriented requirements engineering, which explicitly models problems and their root causes, together with other important ontological concepts, stakeholders' goals, and both functional and non-functional requirements as a solution, and issues. In this framework, ontological concepts are partitioned into layers, reflecting which traceability links are classified into intra- and inter-traceabilities leading to several kinds of links. Additionally, undesirable consequences of inappropriate traceabilities are also categorized. A case study shows some key benefits of the framework.

Index Terms—Requirements Traceability, Ontology, Problem-Aware, Goal-oriented Requirements Engineering

I. INTRODUCTION

Requirements traceability [1] refers to the ability to relate various concepts about requirements, such as problems, goals and requirements, to each other, and helps ensure that the requirements specification is aligned with the intended stakeholders' needs. It aids impact analysis, process visibility, verification and validation, and change evaluation, hence being a critical success factor not only for an initial software development phase but also for its subsequent maintenance phase [2].

Alignment between a requirements specification and the intended stakeholders' needs should be considered in terms of why such needs arise – this in turn in terms of what *problems* the stakeholders are faced with, and what kinds of software system may help alleviate or eliminate the problems.

However, little work can be found on requirements traceability that explicitly considers the problems, even though the key role of a software system is supposedly to help alleviate or eliminate the problems that the stakeholders are suffering from.

In this paper, we propose a problem-aware framework for establishing requirements traceability, in the context of goal-oriented requirements engineering (*GORE*). In this framework, problems and their root causes are explicitly modeled as key ontological concepts (or vocabulary), together with other important ontological concepts, i.e. stakeholders' goals, and both

functional requirements (*FRs*) and non-functional requirements (*NFRs*) as a solution to the problems and at the same time a means for the goals, and issues (e.g., ambiguity and inconsistency) with any such individual concept or any relationship between two individual concepts. In this framework, ontological concepts are partitioned into five layers – problem, goal, requirements, prototype and issue layers. Traceability links then are classified into intra- and inter-traceabilities (layer link). An intra-traceability refers to a relationship between concepts in the same layer, while an inter-traceability refers to a relationship between concepts in two different layers. The layer link can be combined with refinement link (satisficing and decomposition-links), which results in requirements product links for requirements themselves. Additionally, issue link can combine together with refinement link, which leads to requirements process links for rationale of the requirements process. This framework also identifies several classes of undesirable consequences of improper traceability links.

We have carried out a case study, involving about 60 teamwork-oriented course projects in several senior- and graduate-level requirements engineering (*RE*) classes with about 500 students in total. Through this case study, we have observed how the presence of proper traceabilities, or lack thereof, affects the quality of the resulting requirements and prototype. More specifically, our observation shows how frequently omissions or commissions of traceability links are made by students, and of what kinds – leading to incomplete or incorrect requirements. To provide a sense of how this case study has been carried out, a comparison of two groups' work will be shown. A statistical analysis result shows that our framework can indeed help improve the quality of the requirements and corresponding prototypes, by making them more complete, correct and clear, while capturing important rationales.

There has been some work on requirements traceability (e.g., see [1]). Our work is similar to [3], which provides a rich set of issues and ontological concepts such as stakeholder, object, and source as a reference model for requirements traceability, and to [4], which takes a goal-oriented approach to requirements traceability; in our framework, problems and their root causes are treated as a central and fine-grained ontological concept, along with goals, layers and different types of traceability links. The notions of goals and problems are also mentioned in [5], but without ontological layers, layer links (inter-/intra- links), refinement links (satisficing-/decomposition-links), root causes and issues. Model-driven traceability has

been addressed in general [6] and more specifically for requirements in the context of MDA [7], but with different ontological concepts and traceabilities for the life after requirements established; this framework more focus on the life before them.

Section 2 introduces goal-oriented models that have been adopted for the ontology of the framework. Section 3 presents our problem-aware framework for goal-oriented requirements traceability. Sections 4 and 5 respectively describe a case study and a discussion. At the end, a summary of the paper is described, along with some future work.

II. ADOPTED GOAL-ORIENTED MODELS

For our problem-aware traceability framework in the context of goal-oriented requirements engineering (GORE), we adopt work, on the one hand, on representing functional and non-functional requirements, and on representing problems, on the other.

A. Representing Functional and Non-Functional Requirements

Concerning the representation for both FRs and NFRs, there are several goal-oriented frameworks, including KAOS [8], i* [9], and the NFR Framework [10], each with its own emphasis and characteristics. We adopt the NFR Framework, since it is also common to many other goal-oriented models.

An NFR, such as accuracy, security and performance, typically has no clear-cut definition or criteria whether it is absolutely satisfied or not. Accordingly, in the NFR Framework, an NFR is treated as a softgoal, and the notion of “satisficing” is used. Each softgoal has an associated NFR type and one or more topics (See Fig. 4 for examples).

There are three types of softgoals (See Fig. 1): NFR, Operationalizing and Claim softgoals. An NFR softgoal is an NFR to be satisfied; an operationalizing softgoal is a concrete means (e.g., an operation to be carried out by people – expectation, or a FR to be implemented in the projected software system), which can achieve an NFR softgoal; and a claim softgoal is an argument/justification. Each softgoal can be either AND or OR decomposed into sub-softgoals or make a contribution towards satisficing another softgoal, fully or partially positively

(MAKE or HELP), or fully or partially negatively (BREAK or HURT). A bottom-up label propagation mechanism evaluates the effect of a decision on upper softgoals, with a label - Satisfied, Denied, Conflict, or Undetermined. Softgoals and relationships between softgoals are represented in a softgoal interdependency graph (SIG).

B. Representing Problems

There are several different kinds of models for problem representation and root cause analysis, including notably FTA (Fault Tree Analysis) [11], Fish Bone Diagram [12], and PIG (Problem Interdependency Graph) [13]. While FTA is suitable when information is available about AND/OR logical relationships among root causes, Fish Bone Diagram is adequate when uncertainties exist about relationships among root causes. We adopt PIG, since it accommodates both conventions, and additionally it closely resembles SIG, offering NFR softproblem and Operationalizing softproblem, together with the same kinds of satisficing relationships.

III. A PROBLEM-AWARE TRACEABILITY FRAMEWORK IN GOAL-ORIENTED REQUIREMENTS ENGINEERING

Our problem-aware traceability framework in GORE offers an ontology, with problems as among the essential concepts or vocabulary, by extending the two adopted ontological concepts of goals and problems. The framework then introduces five layers of ontological concepts, which in turn lead to several kinds of traceability links. Additionally, the framework offers several different categories of undesirable consequences of omissions or commissions of traceability links.

A. Overall Ontology

Fig.1 shows a somewhat simplified ontology on our adopted goal-oriented models described in the previous section, with detailed refinements of some concepts omitted. It shows key concepts such as problems, goals (including softgoal), requirements, prototypes for requirements visibility and issues for requirements process. As stakeholder is the owner of problems or goals, it represents those problems and goals. As for trace-

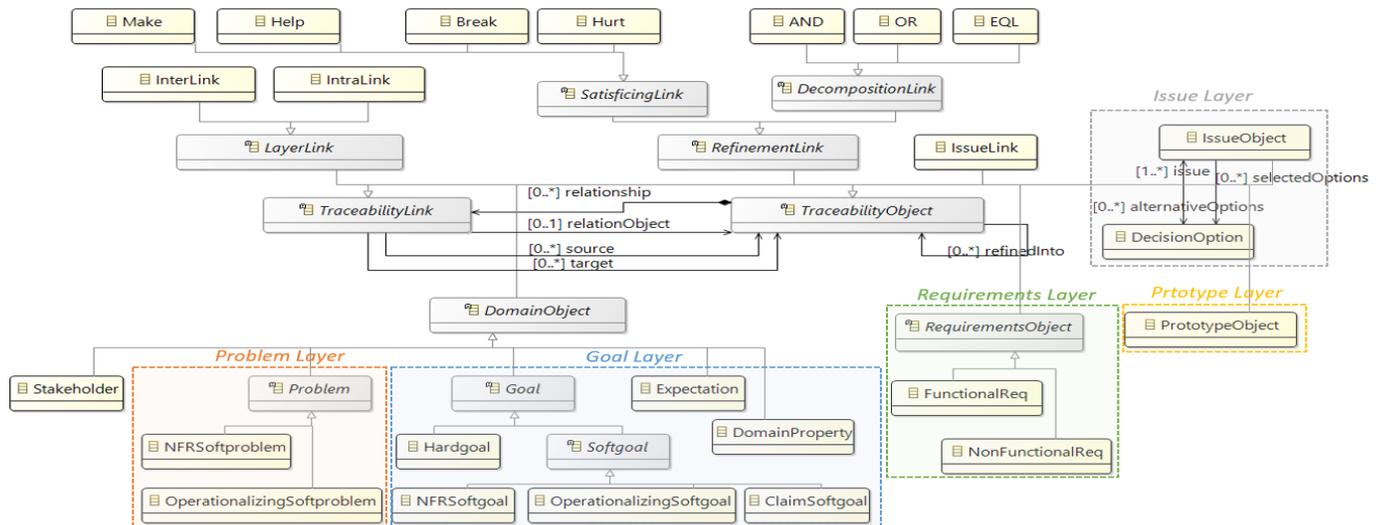


Fig. 1. Ontology for Problem-Aware Traceability in Goal-Oriented Requirements Engineering (GORE)

bility links, there are refinement links such as satisficing- and decomposition- link, and layer link such as inter- and intra-links. Essentially, a traceability link can exist wherever there can be a relationship between the source and target traceability objects. Traceability links are mostly bi-directional, i.e., forward and backward.

B. Layers of Traceability Objects

The ontological concepts are partitioned into five layers, as shown in Fig.1 and Fig. 2, and the example is in Fig. 3.

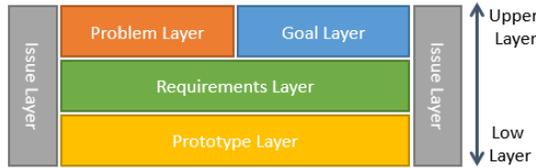


Fig. 2. Layers of Traceability Objects

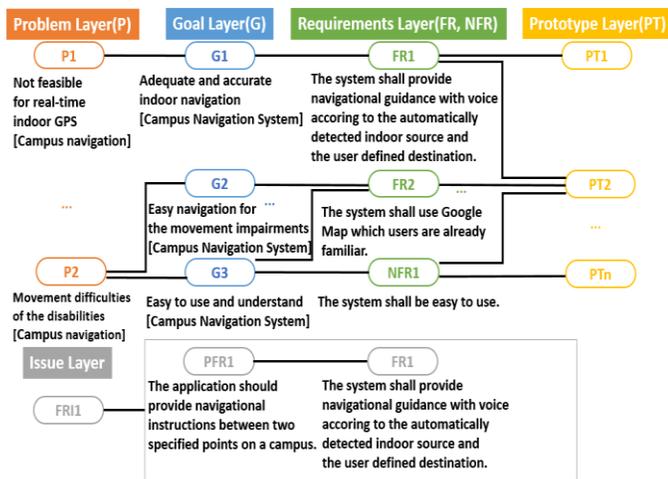


Fig. 3. Examples of Inter Links of Campus Navigation Application for the disabled

- Problem Layer: to represent phenomena that are against stakeholders' goals. This layer is for representing problems and their root causes, which can also help discover critical goals. For example, in Fig. 3, *P1. Not feasible for real-time indoor GPS [Campus Navigation]* is a root cause which handicapped people have in campus navigation. The procedure to extract the root causes is shown in the upper portion of Fig. 4.
- Goal Layer: to represent stakeholders' intentions, which helps to explore alternatives in requirements and select among them. For example, in Fig. 3, *G1. Adequate and Accurate indoor navigation [Campus Navigation System]* is a refined goal, which is against *P1*.
- Requirements Layer: to represent functional & non-functional requirements. For example, *the system shall provide navigational guidance with voice according to the automatically detected indoor source and the user defined destination (FR1)* is a functional requirement, for achieving the goal, *G1*.
- Prototype Layer: to represent a user interface as a prototype which implements FRs and NFRs.
- Issue Layer: to represent issues that happen with any

statement such as ambiguous, inconsistent or conflicting statements, options to resolve them, and trade-off analyses with rationales. More details are discussed in Section C.3.

Relationships between the problem layer and goal layer are generally negative satisficing (e.g., BREAK or HURT). Additionally the requirements layer has positive satisficing relationships (e.g., MAKE or HELP) for the goal layer, as a means, and negative satisficing relationships for the problem layer, as a solution. That's why the problem layer and the goal layer are located in the same layer and requirements layer is under the layers.

C. Classification of Traceability Links

In our framework, there are two kinds of traceability links: Requirements Product- and Requirements Process- Traceability links. While Requirements Product Traceability are links between problems and requirements themselves, Requirements Process Traceability are links for rationales on how the requirements were created.

1) Requirements Product Traceability can be defined as a cross product between Layer Link and Refinement Link:

$$\text{Requirements Product Traceability Link} = \text{Layer Link} \times \text{Refinement Link}$$

1.1) Layer Link: This concerns the boundaries of traceability links, and there are two kinds:

- Intra Link:** refers to any relationship among traceability objects within the same layer. For example, links among NFR, Operationalizing, and claim softgoals are intra links because they belong to the same Goal Layer.
- Inter Link:** any link among traceability objects across different layers (except for the issue layer), including:
 - Problem-Goal (PG) Link: helps ensure all the problems are linked to goals as their context.
 - Goal-Requirements (GR) Link: helps ensure goals are at least partly achieved (satisfied) by requirements.
 - Operationalizing goal-Requirements (OR) Link: helps ensure some operationalizing (soft)goals are linked to functional requirements. All goals should be refined into sub-goals small enough an agent can be assigned. If the leaf goal is assigned to a software system, that will be a requirements. This link shows the relationships between the assigned leaf goal and its requirements.
 - Requirements-Prototype (RP) Link: helps validate the correctness of the requirements, using a prototype.

Fig. 4 shows some intra and inter traceability links within, as well as across, PIG (the upper portion) and SIG (lower portion). Using PIG, abstract problems are further refined, and they can help to find refined critical goals (i.e., leaf goals) by breaking or hurting the leaf problems.

1.2) Refinement Link: This concerns refinement using parent and child relationships:

- Satisficing Link:** refers to any link showing how a traceability object contributes to its parents, through Make,

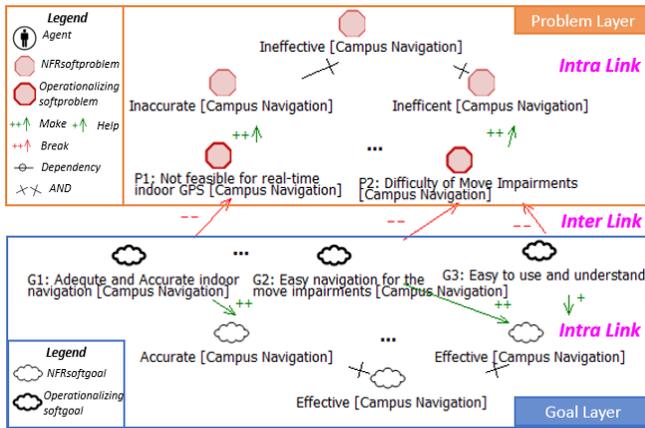


Fig. 4. Examples of Intra and Inter Links

Help, Hurt or Break.

b) *Decomposition Link*: any link showing how a traceability object is composed through AND or OR relationships.

2) *Requirements Process Traceability* can be defined into a cross product between Issue Link and Refinement Link:

$$\text{Requirements Process Traceability Link} = \text{Issue Link} \times \text{Refinement Link}$$

- *Issue Link*

This concerns any link, involving issues (ambiguity, inconsistency or conflict) on anything in any layer, i.e., any problem, goal, FRs/NFRs or prototype features. An issue is associated with options for resolving it, together with rationale for choosing one or more of the options, through tradeoff analysis. Table 1 is an example of issue traceability for a preliminary functional requirement (PFR1), concerning the scope of a campus, which is ambiguous. To alleviate the ambiguity, there are two options and option 2 is selected with a rationale, concerning the project time and resource constraint.

TABLE 1. An Example of Issue Traceability

ID	Description	
FRI 1	PFR1	The application shall provide navigational instructions between two specified points on the ABC campus.
	<i>Ambiguous. Does the campus include outdoor parking lots?</i>	
	Option 1	Define the campus as indoor such as classroom and service buildings.
	Option 2	Include all the campus, from classroom and service buildings to parking lots.
	Choice	Option 1
	Rationale	Given the time and resource constraint of the project, Option 1 is better.

D. Undesirable Consequences of Improper Traceability Links

If traceability links are not properly established or omitted, the following undesirable consequences (i.e. defects) can arise.

- **Incompleteness**: all leaf elements in an upper/same layer are not properly dealt with in a lower/same layer due to the inappropriate forward inter traceability. For example, given a goal “offer a voice job search capability for those with disabilities” in goal (upper) layer, if there are no corresponding requirements to deal

with the goal in requirements (lower) layer, then it will cause incompleteness defects.

- **Incorrectness**: an element in a lower/same layer is not correctly described according to a corresponding element in its upper/same layer due to both improper backward inter and intra traceability links. For example, let us suppose that for the above goal, “the system shall provide a function to learn job skills” is a requirement. The requirement is incorrect because the requirement does not adequately achieve the goal, since learning job skills is not much related to job search.
- **Risk of Gold Plating**: the addition of expensive and unnecessary features to a system due to lack of backward inter and intra traceability.
- **Ambiguity**: there are more than one interpretations and no rationale for a decision because of improper issue links.

IV. CASE STUDY

We have conducted experiments to validate our problem-aware traceability framework, through the group projects of several undergraduate-, graduate-level and industry- requirements engineering courses, which one of the co-authors has been teaching for more than 10 years. Most of undergraduate student were consist of senior student whose majors were computer science, but there were some students who had been working more 10 years in Information Technology (IT) industry. This were similar to the graduate level course. The students of the industry course had IT industry experience more than 3years. We selected projects which were conducted in those courses from year 2011 to year 2015. Most projects were similar domain about building smartphone apps for people with mental or physical difficulties, but not the same apps. There have been about 60 projects which the average number of members were about 8 (12 from senior-level courses, 38 from graduate-level courses, 10 from executive courses for industry people), with about 500 students in total. Students learned several kinds of goal-oriented requirements models, such as PIG or SIG, and applied their knowledge to their projects, using the provided our guidelines and a traceability template which is available [14]. Each project selected the combination of traceability link types on its own decision.

This study has shown the presence of traceability links, or lack thereof, affects the quality of requirements and prototypes. To show the correlation, Teaching Assistant (TA) who is one of coauthors have carried out an overall defect analysis, and analyzed the kinds of defects which students frequently made. To provide a sense of the analysis, we show a comparison between an exemplary project which applied most of the proposed traceability links and another one which did not, which resulted in the functionality differences of their prototypes. Various student documents, which have been used for this case study, are publicly available (including [14]).

A. Overall Defects Analysis

Fig. 5 shows how different kinds of inappropriate traceability links affect the different kinds of defects in terms of the num-

ber of defect projects. The most common kind of defects was requirements incompleteness (as in [15]), and our observation shows that this is caused by omission of traceability links from an upper layer to a lower layer (i.e., forward traceability links). The next frequently-occurring is incorrectness, due to the lack of semantic correspondence between concepts in a lower layer and those in an upper layer that are backward traceable from them.

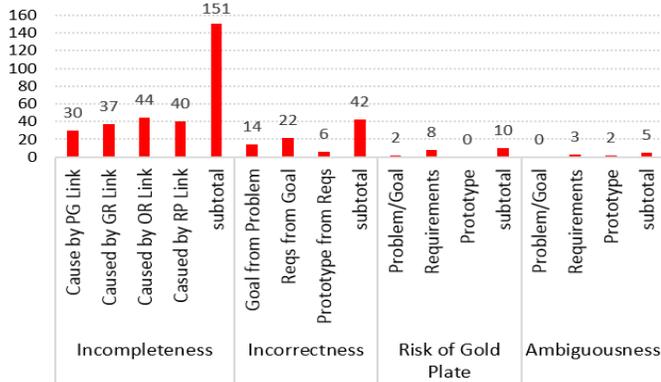


Fig. 5. Overall Defects Analysis

B. Common Defects & Effectiveness of Our Proposal to Reduce Defects

1) Incompleteness

Incompleteness defect, as described in the previous section, arises when all leaf elements of an upper layer are not properly dealt with in a lower layer due to the inappropriate forward inter traceability. This kind of defects mostly result from different inter satisficing links such as Problem-Goal (PG), Goal-Requirements (GR), Operationalizing goal-Requirements (OR) or Requirement-Prototype (RP) and the defects can lead to incompleteness of requirements. To measure the incompleteness defect rate, we divided the target projects into two groups: the projects which applied our suggested link vs. those of not applied each link. Then, for each group, we used the following formula.

$$IncomDR = \frac{IncomDO}{TOT}$$

, where IncomDR is Incompleteness Defect Rate, IncomDO is the number of projects which Incompleteness Defect Occurred and TOT is the TOTal project number, in a specific group.

As Fig. 6 shows, applying inter satisficing links positively affects to reduce the incompleteness defects in all inter link types.

2) Incompleteness

Incompleteness defect arise when elements in a low layer does not semantically correspond to elements in an upper layer due to both improper backward inter and intra traceability links. Similar to the Incompleteness defect measurement, per group, we calculate incorrectness defect rates using the following formula.

$$IncorDR = \frac{IncorDO}{TOT}$$

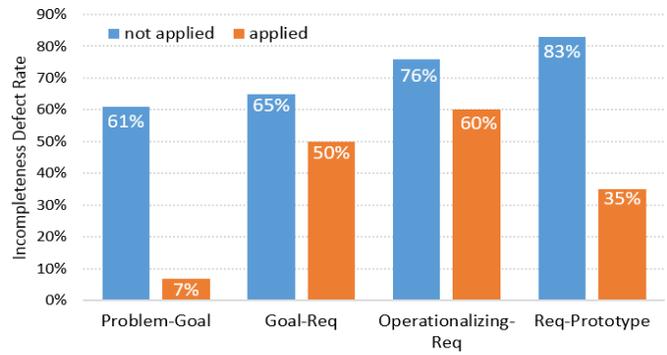


Fig. 6. Incompleteness Rates: When links present vs. absent.

, where IncomDR is Incompleteness Defect Rate, IncomDO is the number of projects which Incompleteness Defect Occurred and TOT is the TOTal projects number, in a specific group.

Fig.7 shows that the comparison of incorrectness defects between when the intra and inter links are applied and when they are not applied. This also showed positive effects on reducing incorrectness defects.

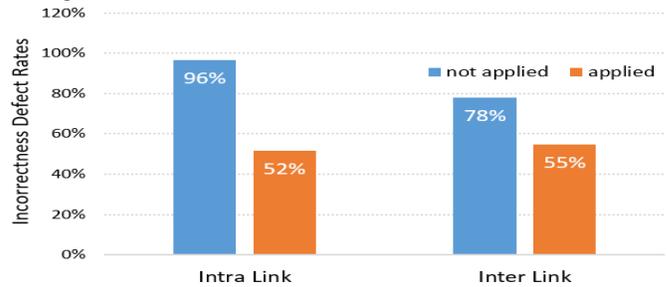


Fig. 7. Incorrectness Rates: When links present vs. absent.

C. Projects Comparison

In order to provide a sense of our case study, a comparison is given of two projects on the same subject - here, medication alert to help remind those elder people who often forget to take their medications. As Table 2 shows, while Team A applied most of the proposed links, including inter, intra and issue, Team B applied only Issue Link. Consequently, as Fig. 8 shows, while team A has more detailed information, such as Caregiver Notification and Refill Reminder which can help elder people in diverse ways, team B only has time setting.

TABLE 2. Traceability Links Applied by Two Teams

Link Type		Team A	Team B
Inter	Problem-Goal	O	X
	Goal-Requirements(Req)	O	X
	Operationalizing-Req	O	X
	Req-Prototype	O	X
Intra	Problem	O	X
	Goal	O	X
Issue Link		O	O

V. DISCUSSION

A. Overall Observation: Although many students knew each concept, such as problem, goal, requirements and prototype, they did not well recognize relationships between problems and goals, goals and requirements, or operationalizing softgoals and

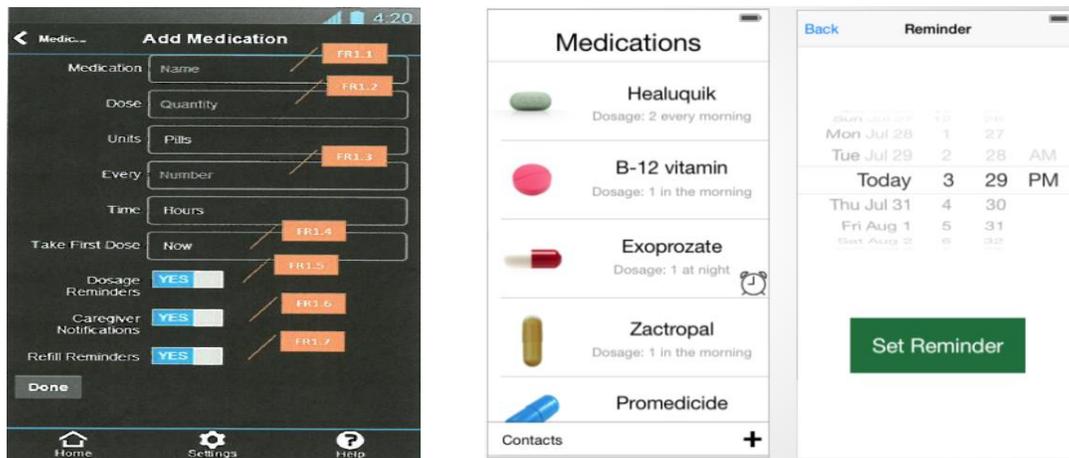


Fig. 8. Prototypes of Team A (left black background) vs. Team B (right white background)

functional requirements. Overall, the various kinds of traceability links we provided helped the students (better) understand such relationships. Through our case study, we also have observed that such links help reduce requirements defects, hence improving the quality of the requirements.

B. Threats to Validity: The quality of requirements depends on the manpower and capabilities – both individual and teamwork skills - of project team members. In our experiment, each team had different members and also the team size varied from one team to another. Moreover, the project scopes were diverse, although all of them had to do with a smartphone app for people with mental or physical disabilities, hence our experiment was not entirely homogeneous. Also, the sample size for this case study was small for a significant statistical analysis, concerning scalability, although the total number of students who participated was about 500. Additionally, the analyzer may not fully understand the each project.

VI. CONCLUSION

In this paper, we have proposed a problem-aware approach to requirements traceability in goal-oriented requirements engineering, in order to help ensure that the requirements specification and its corresponding prototype are well aligned with their intended stakeholders' needs. More specifically, this paper has presented 1) an ontology, which incorporates such key concepts as problems, goals, requirements, prototypes and issues; 2) five layers, which partition the ontological concepts, according to their levels of abstractions; 3) several classes of traceability links, which group relationships, according to the semantic closeness of the ontological concepts; and 4) several classes of undesirable consequences of improper traceability links. A case study, we feel, shows that our approach helps develop better requirements specifications and prototypes, and importantly in a traceable manner. This study, we feel, also has shown that our approach facilitates the detection of several different kinds of undesirable consequences of inappropriate traceabilities, such as incompleteness, inconsistencies and ambiguities in the requirements specifications and the prototypes – these defects will likely result in an implemented software system with the same kinds of defects.

There are several lines of future work. One is regarding (semi)-automatic mapping between layers and detection of

omission or commission of traceabilities by using MDA (Model Driven Architecture). Another line of future research concerns provision of better templates for the various kinds of traceabilities. More case studies are also needed in various types of application domains.

REFERENCES

- [1] O. C. Z. Gotel and A. C. W. Finkelstein, "An Analysis of the Requirements Traceability Problem," *Proc., 1st Int. Conf. on Reqs. Eng.*, 1994. pp. 94-101.
- [2] B. Ramesh, "Factors Influencing Requirements Traceability Practice," *Communications of the ACM*, 1998. pp. 37-44.
- [3] B. Ramesh and M. Jarke, "Toward Reference Models for Requirements Traceability," *IEEE Trans. Soft. Eng.*, 2001. pp.58-93.
- [4] J. Cleland-Huang, R. Settini and O. Benkhadra, "Goal-Centric Traceability for Managing Non-Functional Requirements," *Proc. 27th Int. Conf. on Software Engineering*, 2005. pp.362-371.
- [5] M. Yamin, V. Zuna and M. A. I. Bugami, "Requirements Analysis and Traceability at CIM Level," *Jour. of Soft. Eng. and App.*, 2010. pp. 845-851.
- [6] N. Aizenbud-Reshef, B. T. Nolan, J. Rubin and Y. Shaham-Gafni, "Model Traceability," *IBM Sys. Jour.*, 2006. pp. 515-526.
- [7] A. João Paulo A, M.E. Jacob and P. van Eck. "Requirements Traceability in Model-Driven Development: Applying Model and Transformation Conformance," *Inf.Sys.Fron.*, 2007. pp. 327-342.
- [8] A. van Lamsweerde, "Requirements Engineering in the Year 00: A Research Perspective," *Proc., 22nd Int. Conf. on Soft. Eng.*, 2000. pp. 1-15.
- [9] E. Yu, P. Giorgini, N. Maiden and J. Mylopoulos, *Social Modeling for Reqs. Eng.*, The MIT Press. 2011.
- [10] J. Mylopoulos, L. Chung and B. Nixon, "Representing and Using Nonfunctional Requirements: a Process-Oriented Approach," *IEEE Trans. on Soft. Eng.*, 1992. pp. 483-497.
- [11] L. Xing and S. V. Amari, "Fault Tree Analysis," *Handbook of Performability Engineering*, Springer London, 2008. pp.595-620.
- [12] K. Ishikawa, *Guide to Quality Control*. No. TS156. I3713, 1982.
- [13] S. Supakkul and L. Chung. "Extending Problem Frames to Deal with Stakeholder Problems: An Agent- and Goal-Oriented Approach," *Proc., ACM Sym. on App. Com.*, 2009. pp. 389-394.
- [14] <http://www.utdallas.edu/~chung/RE/syllabus.htm>
- [15] T. Clancy, "The Standish Group Report," *Chaos Report*, 2014.

Approach to Define a Non-Functional Requirements Elicitation Guide Using a Customer Language

Andreia Silva, Placido Pinheiro, and Adriano Albuquerque

Graduate Program in Applied Informatics
University of Fortaleza (UNIFOR)
Fortaleza, Brazil

andrearsp@gmail.com, {placido,adrianoba}@unifor.br

Jonatas Barroso

Computer Department
Federal University of Ceara (UFC)
Fortaleza, Brazil
jonatasbarroso@gmail.com

Abstract—Non-functional requirements (NFR) have a crucial role in the software development process because they correspond to the characteristics and restrictions on which the software must running and represent factors that influence the time and cost of software development. Nevertheless, many organizations do not perform these requirements elicitation properly. This paper presents an approach to creating a NFR elicitation guide focused on customer language. To create the approach was performed a systematic review which identified and analyzed related works. Also a survey was conducted which helped to know the current situation of NFR elicitation in software development organizations and obtained suggestions for composition of the NFR elicitation guide. Finally, the results of applying the proposed approach presents evidence that the use of the process is feasible and produces better quality requirements.

Keywords—process; approach; elicitation; non-functional requirements; NFR; customer

I. INTRODUCTION

All systems have non-functional requirements (NFR) but are not always explicitly defined in a formal specification [1]. NFR elicitation is a complex task and some factors contribute directly to this. Firstly, NFR are very diverse. This makes it difficult an analyst know all applicable kinds to a given software context. Customers, in turn, do not always know their software non-functional needs or do not know how to explicit them. In addition, the several knowledge sources, such as standards and norms or relevant bibliographic references related to NFR, do not specify in what situations a requirement should be elicited or define a pattern for its definition. Furthermore, while there have been proposed works dealing NFR elicitation, yet there is no evidence that indicate the most appropriate method for these requirements eliciting. None of these methods has been adopted as standard by the requirements engineering community [2].

II. METHODOLOGY

Conducting a systematic review of over 1700 publications identified related studies with the NFR elicitation. The main research question was identify approaches related with guides utilization to support the NFR specification. The search was conducted in digital libraries of ACM, IEEE and Scopus, including top conferences interested in requirements

engineering studies. Then many papers that define procedures for NFR elicitation were analyzed and only two of these publications were considered related works.

Kopczynska and Nawrocki [3] present results of a case study on NFR elicitation through a method based on the knowledge reuse. This method consists of a short sessions sequence to discuss the ISO 25010 [4] characteristics. In each session are used predefined requirements with values to be replaced by analysts during the specification. New templates can be created from the NFR elicited and reused in other software projects. On the other hand, Balushi [5] propose a framework for NFR elicitation supported by quality requirements ontologies defined from the application context to be developed. The specification process is aided by a tool that contains a knowledge base of requirements already elicited. Each new project updates the database with new requirements identified.

In addition, a survey was held in order to get more information about the NFR elicitation in software development organizations. In summary, main objectives of the research were: (i) an overview of the current situation related to the practice of NFR elicitation and (ii) identify the reasons for these requirements are not elicited. This survey counted with the participation of 100 professionals representing several roles involved in software development. About 69.4% of respondents said elicit NFR in their companies. However, only 11.8% of these professionals consider that the NFR definition is good. Participants also reported the reasons why these requirements are not elicited often. The main reasons for this failure definition are: lack of knowledge of the teams, lack of request of these requirements by the customer, and lack of technical or financial organization capacity to comply with NFR. The survey also sought to evaluate the need for a guide to support the NFR elicitation. About 90% of participants said that a guide would be useful to identify non-functional requirements and half of all respondents reported suggestions for the contents of this guide.

III. APPROACH TO CREATING AN ELICITATION GUIDE

A literature systematic review provided an understanding of the main approaches related to the NFR elicitation. The survey, in turn, allowed to identify the expectations of the involved professionals in the software development process about what

should be addressed in the approach proposed in this paper. Thus, these steps contributed to define the proposed process. This approach is performed before the beginning of any organization's project and aims to create a NFR knowledge database that can be reused in projects. Thus, it translates the expert knowledge in the most appropriate questions and requirements to a better understanding of customers and business analysts, allowing disseminate this knowledge with other organization members. A preliminary version of this approach can be seen in [6]. This paper presents the evolved version, including details of how to perform each of the proposed activities and the results of user experience.

A. Approach Description

Fig. 1 shows an approach overview detailed in the following subtopics.

1) *Define Software Type*: The first process activity involves defining the software type for which you want to create the elicitation guide. Examples of software type classification can be found in [7] and [8]. To perform the choice of the corresponding type, the organization can analyze the developed products and seek to identify applications that require more attention in terms of NFR or simply select the application type that develops more frequently.

2) *Select Reference Basis*: This activity involves choosing a bibliographic source to guide the definition because it is not always easy to distinguish functional and non-functional requirements. Also, knowing many NFR types helps to define which type is important for the software that the organization develops. For that, some important references are available in the literature. Among the most relevant NFR catalogs is the list provided by Chung [9] which contains more than one hundred requirements types. Another requirements types classification is presented in ISO 25010 [4] standard and Sommerville [10].

3) *Identify Experts*: This activity consisting in identifying the organization professionals working in technical areas related to the NFR types. In some cases, the organization knows the most important requirements types for its context, in other cases may not know anything about these requirements. Thus, the responsible to implementation approach can make a prior analysis of requirement types addressed in reference database and identify experts available in the organization. These professionals will participate in this process activity until choose the requirements to be treated.

4) *Select Requirements Types*: After defining the reference database that will guide the requirements definition process and identifying the organization professionals who participate of this process, you must select the requirements types that will be part of work scope. The applicability of each requirement type for the software defined context should be evaluated in this step of approach.

5) *Identify Questions*: The next step involves analyzing each NFR type and verifying how the customer can be asked about the software requirements. This is one of the most

important process activities, because it seeks to obtain a complex information in a simple and comprehensible to a user without technical profile way.

6) *Define Requirements*: Then, for each identified question should define requirements models, establishing clearly the customer needs and allowing the analyst to get the parameters for the elicited requirement. The attributes that must be defined for each requirement are shown in Table I.

7) *Validate With Experts*: This validation aims to verify if the requirements are understandable and relevant to the technical areas. In general, technical professionals need the information contained in the NFR to perform their jobs. Therefore, ensure that defined requirements meet this purpose is important.

8) *Validate With Users*: Then, the requirements must be evaluated by users; in this case anyone who has contact with the customer in the requirements elicitation activity. This evaluation aims to ensure that the requirements are suitable for the customer language, unlike the previous evaluation, which verifies that the requirements are feasible from a technical point of view.

TABLE I. GUIDE REQUIREMENTS ATTRIBUTES

Attribute	Description
ID	Unique requirement identifier in the guide.
Requirement Type	Requirement class to which the NFR belongs.
Concepts	Requirement type definition, described preferably based on norms, standards, technical articles, books or other recognized references.
Question	Question asked to the customer about the corresponding requirement. Responses should be simple and direct to avoid subjective interpretation.
Template	Requirement writing pattern, including the parameters. There are two parameter types: mandatory, delimited by < and >, and optional enclosed in [and].
Example	Requirement based on the established model including real values for the variables. We suggest that the example values are defined based on the organizational service capacity, because, this way, business analysts have the information than can be met by the organization at the time of negotiation with customers.
Mandatory	Indicates if the requirement is required or desirable.
Dependency	Indicates which other requirements should be defined.

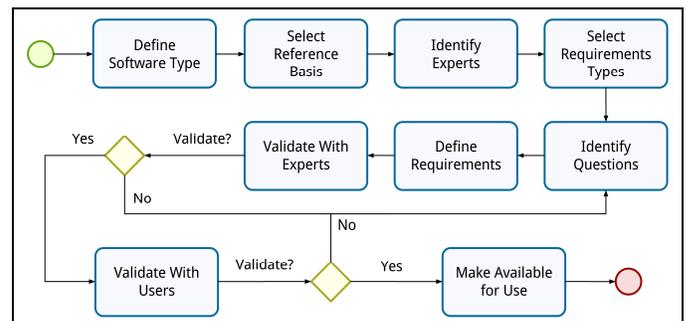


Figure 1. Process for Creation of the Elicitation Guide

9) *Make Available for Use*: Finally, after evaluation activities, the bidding guide should be available for use during the software requirements identification stages. The professionals involved in the requirements elicitation process must be notified of guide availability. Note that the guide can be used as a support tool in any requirements elicitation process.

IV. APPROACH USE EXPERIENCE

In order to analyze the proposed approach feasibility, was held an use experience in an organization that provides IT and communication solutions for the Brazilian government. This organization has a defined process for requirements elicitation but there is no specific guideline to NFR. Approach application was conducted between July and September 2015.

A. Approach Execution

The following subsection describe how the execution of each activity that compose the approach was performed.

1) *Define Software Type*: During the discussions has been decided that web applications would be the software type to be treated. Although developing other applications types, most of organization customers requests involves this software type. After definition, managers requested to prioritize the requirements relating to security, performance and usability because, being a public organization, some standards published recently required the treatment of these requirements.

2) *Select Reference Basis*: During the activity execution, the NFR types sources were presented. Among them, the management team has selected as a reference the ISO 25010 [4] characteristics catalog. In addition to being an international standard, this catalog has a well-defined classification of requirement types that the organization had interest in dealing.

3) *Identify Experts*: For the experiment, the organization provided a group of professionals who perform activities in different technical areas. The choice of professionals occurred considering the performance of them in technical areas related to NFR. Experts were selected the areas of architecture, security, capacity and availability, and performance, among others. No less important, all selected professionals have over 8 years experience in IT and at least 3 years working in the organization. Regarding the academic level, 18% are masters, 45% specialists and 36% graduates.

4) *Select Requirements Types*: Choosing requirements types occurred by analyzing each subcharacteristic presented by ISO 25010 [4]. During this activity, experts evaluated each requirement type from the perspective of importance to the organization context who work and software selected type. No subcharacteristic related to Functional Suitability and Maintainability were selected for treatment. According to experts, the organization has strict contracts that do not allow variations in the compliance degree the defined scope for the project, unless there is change negotiations this scope. For this

reason, the Functional Suitability characteristic was not selected. Regarding Maintainability, experts reported that the organization works with its own development framework that already includes a lot of good practices that seek to ensure a good maintainability level and therefore not need to establish such requirements. During the discussions, the team identified non-functional requirements that should be elicited but that did not fit into any of the standard characteristic. Then, three new subcharacteristics related to Technological Constraints and Legality have been added to the identified types list. Although not addressed by ISO 25010 [4], are cited by Sommerville [10] and were included in the scope of work because of the organization needs.

5) *Identify Questions*: After selecting the requirements types, each subcharacteristic was analyzed from the perspective of how the customer could be asked if the corresponding requirement would apply to the software. The proposed questions were evaluated rigorously to ensure that only "Yes" and "No" answers were possible. This was done to maintain a standard and ensure simple and direct questions. These questions aimed at capturing the requirement necessity. Other information required for the requirement are treated as attributes in the next step.

6) *Define Requirements*: This step was conducted with the expert support who were consulted to answer questions concerning the requirements applicability and about the organization indicators. Obtained values were needed to set requirement examples that should be defined based on organizational service capacity. Moreover, in certain cases, the organization does not keep important information concerning the capacity. Therefore, requirement definition that depend on this information is a motivating factor for the organization understand better its performance. For example, to define recoverability software requirements after a failure, the organization must know the time when the services were restored. This is not always known and the requirement definition alerts the organization to this fact. In addition, for each question at least one NFR model was suggested such that, when the customer answering it, the business analyst has predefined requirements to be elicited. Thus, 36 questions and 49 non-functional requirements have been defined. Table II shows NFR performance examples obtained with the implementation of the approach.

7) *Validate With Experts*: After requirements defining, each was analyzed using some criteria (e.g., completeness, clarity, relevance, and capacity to implement and test) and the comments were recorded and treated. This activity repeated until all requirements groups were fully validated.

8) *Validate With Users*: We selected three business analysts for this stage. All received, in addition to the elicitation guide, a brief orientation about the work purpose and use of the guide. Then users evaluated the guide and requirements according understandability and usability criteria. The evaluation result was satisfactory because only

five comments were made about the clarity of few requirements. Thus, some improvements were made after the evaluation. However, it was noticed that the basic understanding of some technical issues are needed and training on these issues is essential to avoid possible doubts on the customers questions. Finally, all analysts said the guide is easy to learn and use. One user also reported that the guide would greatly help for the work performed by them.

9) *Make Available for Use*: After validation of experts and users, the NFR elicitation guide was made available to support the organizational process. An email informing about the guide was sent to all business and systems analysts.

V. CONCLUSION

Some related work have mechanisms to support the NFR elicitation. The proposed approach differs from these work because adds some points that are not covered by other studies. It is a process for a elicitation guide creation performed before the start of any project. With this, the organization has the ability to analyze in detail the technical issues relevant to the context of software developed by it. Thus, the customer will be asked about relevant points for the architecture, infrastructure and other key issues for software development from the early stages of the project. In addition, the approach defines requirement examples with parameters filled based on the organizational service capacity. Also provides the evaluation of the requirements quality, not only from a technical perspective, but also from the point of view to the customer language adequacy.

In addition, the approach presents concepts of each requirement type in order to facilitate the understanding of the business analysts. Therefore, along with the issues focused on the customer, it becomes an easy guide to learn and use in the requirements elicitation process. Once the guide is constructed outside the projects scope, enables optimizing the experts time who will not have to attend during the elicitation process of each developed project in the organization. In addition, an organization that does not have expert professionals in all NFR types can count with support of a consulting for the approach implementation and then enjoy the results generated, i.e., reusing this knowledge database to support elicitation of all your projects.

The result obtained using the approach was compared with elicited requirements without the use of the approach in other organization projects. It could be observed that the proposed approach contributed significantly to the NFR elicitation in terms of quantity, quality and diversity of the requirements.

The proposed approach has as main innovation integrate the customer in NFR definition process in order to promote a change of the existing culture currently, where it is believed that many requirements types cannot be obtained from the customer. To deal with this issue, the approach works the translation of specialized knowledge in questions and more appropriate requirements for customers and business analysts. In addition to allowing disseminate the knowledge of experts with other organization members.

TABLE II. REQUIREMENTS EXAMPLES USING THE APPROACH

Type	Time Behavior
Definition	Degree to which a product or system performs its functionalities according to the specified response time, processing time and throughput.
Question	There is some functionality that are necessary to determine the response time?
ID	4.1.1.1
Template	The system should have the following limits of response times when exposed to load defined by Requirement 4.1.2.1, which deals with the number of transactions: <ul style="list-style-type: none"> <operation>: up to <response time> in <percentage of acceptance> of the requests.
Example	The system should have the following limits of response times when exposed to load defined by Requirement 4.1.2.1, which deals with the number of transactions: <ul style="list-style-type: none"> Digital document upload: up to 2s in 90% of the requests.
Mandatory	Yes
Dependency	4.1.2.1 become mandatory.

Application of the proposed approach has identified some improvement opportunities and a third approach version is being built including other factors, such as identifying ways of measuring the size of these requirements.

REFERENCES

- [1] J. Slankas and L. Williams, "Automated extraction of non-functional requirements in available documentation," proceedings of the 1st International Workshop on Natural Language Analysis in Software Engineering (NaturaLiSE), San Francisco, CA, pp. 9-16, 2013.
- [2] M. M. Rahman and S. Ripon, "Elicitation and modeling non-functional requirements - a pos case study," proceedings of the International Journal of Future Computer and Communication, vol. 2, pp. 485-489, 2013.
- [3] S. Kopczyńska and J. Nawrocki, "Using non-functional requirements templates for elicitation: A case study," proceedings of the 2014 IEEE 4th International Workshop on Requirements Patterns (RePa), Karlskrona, pp. 47-54, 2014.
- [4] ISO/IEC 25010:2011, Systems and software engineering/Systems and software Quality Requirements and Evaluation (SQuaRE)/System and software quality models, 2011.
- [5] T. H. Al Balushi, P. R. F. Sampaio and P. Loucopoulos, "Eliciting and prioritizing quality requirements supported by ontologies: a case study using the elicito framework and tool," in Expert Systems 30, no. 2, pp. 129-151, 2013.
- [6] A. Silva, P. Pinheiro, A. Albuquerque and J. Barroso, "A Process for Creating the Elicitation Guide of Non-functional Requirements," proceedings of the 5th Computer Science On-line Conference 2016 (CSOC2016), vol. 2, pp. 293-302, 2016.
- [7] D. Mairiza, D. Zowghi and N. Nurmuliani, "An investigation into the notion of non-functional requirements," proceedings of the 2010 ACM Symposium on Applied Computing, pp. 311-317, 2010.
- [8] A. C. Guerra and R. T. Colombo, "Information technology: software product quality," "Tecnologia da informação: qualidade de produto de software," Science and Technology Ministry, Brasília, DF, 2009.
- [9] L. Chung, B. A. Nixon, E. Yu and J. Mylopoulos, "Non-functional requirements in software engineering," proceedings of the International Series in Software Engineering, vol. 5, p. 476. Springer, Heidelberg, 1999.
- [10] I. Sommerville, "Software Engineering," Addison-Wesley Longman Publishing Co., Boston, MA, 2006.

A Characterization of Negative User Stories

Pankaj Kamthan
Concordia University
Montreal, Canada
kamthan@cse.concordia.ca

Nazlie Shahmir
WestJet Airlines Limited
Calgary, Canada
nshahmir@westjet.com

Abstract—In the context of an agile project, negative interactions are addressed by equipping the ‘conventional’ positive user story engineering process with a number of conceptual models, including those for negative user story and negative role. The challenges inherent in eliciting negative uses, negative roles, and negative user stories are highlighted. The cost of engineering negative user stories is analyzed. The relationships among positive and negative user stories are considered. For illustration, a detailed example is presented.

Keywords—agile methodologies; conceptual modeling; requirements engineering; risk management; security engineering; socio-technical distributed systems

I. INTRODUCTION

The E-Type software systems are a large class of systems that reflect human processes or the real world [1]. In recent years, the agile methodologies are being used increasingly in the industry for the development of E-Type software systems aimed for general public consumption, such as distributed software systems, in general, and interactive Web Applications, in particular [2].

If an agile project team anticipates a positive *user experience* (UX), then potential negative uses and their impact need to be *forethought* and need to be essential concerns throughout the agile process. The purpose of this paper is to serve as a starting point towards such an endeavor by proposing a ‘network’ of conceptual models that provide necessary abstraction and input to different steps in modeling negative users and potential negative uses to support the positive *user story engineering* process.

The rest of the paper is organized as follows. In Section II, background is provided and related work is highlighted. The elements of negative user story engineering are introduced in Section III. In Section IV, directions for future research are outlined. Finally, in Section V, concluding remarks are given.

II. BACKGROUND AND RELATED WORK

The negative uses of software systems predate the Internet, but have been catalyzed by the broad acceptance by general public and use of the Web for activities beyond for which it was envisioned originally. The negative uses can be different kinds. Traditionally, the notion of negative use has been related intimately to *privacy* (about preventing psychological harm), *security* (about preventing possessional harm), and/or *safety* (about preventing physiological harm).

The impact of such negative uses ranges from *innocuous* (annoying and/or distracting) to *nocuous* (cognitively, emotionally, financially, legally and/or socially damaging), temporarily or permanently. The negative uses seriously undermine the otherwise increasingly important role played by software for the benefit of society.

In agile methodologies, software requirements are usually expressed as either *user stories* or lightweight *use cases*. In the rest of the paper, ‘user story’ and ‘positive user story’ are considered synonymous and used interchangeably.

In the past decade or so, addressing negative use during (agile) requirements engineering has garnered much interest. For example, the notion of *abuse case* [3] and the concept of (and graphical modeling notation for) *misuse case* [4] have been proposed and have been used for the elicitation of security requirements. XP practices have been extended to support the creation of security-related user stories that are informed by a risk assessment of *abuser stories* [5]. However, the attention on the abuser and the form for expressing abuser stories is inadequate. There is preliminary work on modeling negative user stories [6], which, in part, forms the motivation for this paper.

III. ELEMENTS OF NEGATIVE USER STORY ENGINEERING

A. Conceptual Models

The compendium of conceptual models that follows is a requisite for a comprehensive understanding and systematic engineering of negative user stories. These models, of which some appear in an earlier work [6], are intrinsically interrelated and are informed by international standards.

1) Context of Use Model

This model provides an understanding of the technical as well as non-technical *environment factors* under which a user uses the software system. The technical factors include network connection type, device type, and operating system type. The non-technical factors include mental and physical ability of the user. For example, the ISO 9241-210 Standard can be a source for such a model.

2) Positive User Story Model

This model provides an understanding of the notion of positive user story, and highlights the elements necessary for expressing a positive user story properly [7]. For uniformity, a positive user story statement could be structured in controlled natural language text as: A role can goal to value. The goal must be explicit; the value may be implicit or explicit.

3) Positive Role Model

This model provides an understanding of the characteristics and behavior of a typical positive user (playing a particular role) [6]. For example, a *persona* (an archetypical user of a software system) can be such a model [8]. It helps create empathy among requirements engineers towards positive users.

4) Negative Use Model

This model provides an understanding of the types of negative uses that a software system could be subjected to, their probabilities of occurrence, and their consequences, if realized [6]. It helps create awareness of negative uses among requirements engineers, and provides the knowledge (including terminology) necessary for expressing a negative user story properly.

5) Negative User Story Model

This model provides an understanding of the notion of negative user story, and highlights the elements necessary for expressing a negative user story properly [6]. For uniformity, a negative user story statement could be structured in controlled natural language text as: A negative role wants to negative goal to negative value. A negative user story is not designed or implemented. It therefore has no acceptance criteria or estimate, but is associated with a risk assessment.

6) Risk Assessment Model

This model provides an understanding of risk assessment to be able to make informed decisions about negative user stories. According to the ISO Guide 73, a *risk source* is the element that has the intrinsic potential to give rise to risk. For example, unprotected credit card information of a customer is a risk source. A *risk* is the combination of the probability in the interval (0, 1) of a *threat* (a circumstance with the potential to produce loss) and its *consequence* (the loss that will be incurred if the corresponding threat is realized). A *risk exposure* (RE) is the potential loss incurred by a risk.

RE is a function of the likelihood of the threat and the impact of its consequence. Using these as the two orthogonal dimensions, RE can be given qualitatively by a *risk matrix*, as shown in Fig. 1. In this case, the possible RE levels are N (Negligible), L (Low), M (Medium), H (High), and E (Extreme). RE can serve as a basis for *risk assessment*. For example, RE level of “M”, “H”, or “E” could be seen as significant, whereas RE level of “N” or “L” could be seen as insignificant.

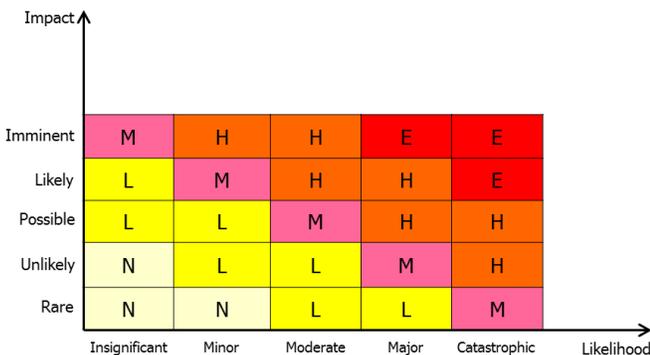


Figure 1. A risk matrix.

7) Negative Role Model

This model provides an understanding of the characteristics and behavior of a typical negative user (playing a particular role) [6]. For example, a *negative persona* (an archetypical negative user of a software system) can be such a model [8]. The negative roles can be of different kinds, including *Fraudster*, *Imposter*, *Malicious Hacker (Black Hat Hacker)*, *Phisher*, *Prankster*, *Spammer*, and *Vandal*. There is currently no standard classification of negative roles. The actions of a negative role are unethical, but may or may not be unlawful. For example, the actions of a black hat hacker are normally unlawful, but that of a prankster are not.

B. Challenges in Eliciting Negative Uses, Negative User Stories, and Negative Roles

The development of a software system can be viewed as acquisition of knowledge throughout the (agile) process. In a model of knowledge (or lack thereof, that is, ignorance) of a person, there are a number of different states in the increasing order of ignorance [9]:

- **Zero-Order Ignorance (0OI)—Lack of Ignorance:** “A person knows something.” For example, a software engineer knows about a vulnerability in the application programming interface (API) of a programming language used in the implementation.
- **First-Order Ignorance (1OI)—Lack of Knowledge:** “A person knows that he or she does not know something.” For example, a software engineer knows that he or she does not know the motives of a negative role or the time a negative role may interact with the software system, or the irreparable financial loss or the emotional impact that can incur on the customers due to negative use.
- **Second-Order Ignorance (2OI)—Lack of Awareness:** “A person does not know that he or she does not know something.” For example, a software engineer does not know that he or she does not know about the criticality of certain security-related defects missed during testing.

To ‘think negatively’ and be able to anticipate all possible negative uses of a software system can be difficult for software engineers, even if they are trained in critical thinking.

The candidate negative roles and the candidate negative user stories could, as usual, be identified, classified, and prioritized, using *ideation techniques* (such as brainstorming and mind mapping). However, certain *ethnographic techniques* (such as interviews and surveys), which have proven to be useful for positive roles and positive user stories, cannot be applied effectively for negative roles and the candidate negative user stories.

It may be difficult to prevent negative uses entirely [2]. Indeed, with the evolution of the Social Web and increasing use of distributed software systems, the inception of new types of negative uses (such as social engineering) is inevitable and the number of negative uses is unlikely to decrease. Even if a negative role’s motives or ways may be unfamiliar (due to 1OI or 2OI), it can be expected that he or she exploits vulnerabilities underlying familiar aspects of

software, such as boundary conditions, intersystem communication, and system assumptions. For example, in the context of a Web client-server environment, if HTTP cookies are relied on exclusively for user identification and if it is assumed that the Web client never modifies its HTTP cookies before they are sent back to the requesting Web server, then a negative role could cause problems by taking control of the session and modifying the HTTP cookies.

C. A User Story Engineering Process for Negative Uses and its Implications

The ‘conventional’ positive user story engineering process needs to be *extended* to accommodate negative user stories, as shown in Fig. 2. The conceptual models introduced in the previous section are an input to this process.

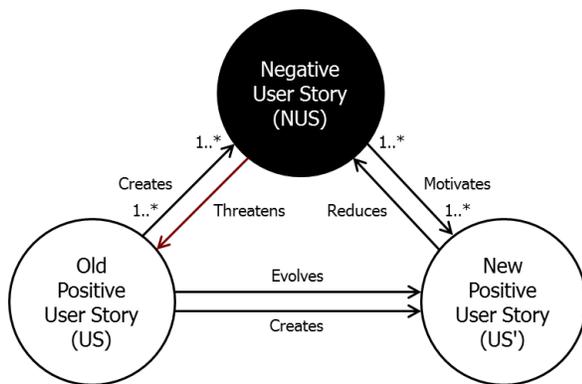


Figure 2. A negative user story influences a positive user story.

The dynamics of the process in Fig. 2 can be explained as follows. Let there be a positive user story, US. The existence of US creates the potential for a negative user story, NUS, which, in turn, threatens a successful realization of US, and therefore prompts a change to the US. If a risk assessment concludes a significant RE level of NUS, then there are two options for *reducing* the risk posed by NUS: (1) evolve old positive user story, US to US', and/or (2) create new positive user story, US'. For example, (1) could involve refining the text of US and/or splitting US (into two or more user stories).

D. Economics of the User Story Engineering Process for Negative Uses

The addition of a negative user story in the ‘conventional’ positive user story engineering process entails cost that, for the sake of feasibility, need be weighed against the perceived benefits. There are different kinds of costs associated during the positive user story engineering process that involves a negative user story:

1. There is cost, say C_1 , if the negative goal and negative value of the negative user story are satisfied. This cost could, for example, be measured in terms of customer dissatisfaction.
2. There is cost, say C_2 , of risk assessment.
3. There is cost, say C_3 , of evolving old positive user stories or eliciting new positive user stories, as the case may be.

The probability of incurring C_1 belongs to a continuous set, whereas the probabilities of incurring C_2 or C_3 belong to a discrete set.

There are three cost scenarios:

Cost Scenario 1: If there is no risk assessment, then there is no need for evolving old positive user stories or eliciting new positive user stories. In this case, there is no C_2 or C_3 , but there is still C_1 . In this case, it is assumed that $C_1 < C_2 + C_3$.

Cost Scenario 2: If a risk assessment suggests an insignificant RE level, then there is no need for evolving old positive user stories or eliciting new positive user stories. In this case, there is no C_3 , but there is still C_1 and C_2 . In this case, it is assumed that $C_1 + C_2 < C_3$.

Cost Scenario 3: If a risk assessment suggests a significant RE level, then there is a need for evolving old positive user stories or eliciting new positive user stories. In this case, there is C_2 and C_3 , and, hopefully, $C_2 + C_3 < C_1$.

E. Example

Let there be an electronic shopping system being developed as a Web Application using one of the agile methodologies that has inherent support for user stories. This shopping system may typically have a shopping cart system, a payment system, and a customer identity management system.

The existence of a positive user story, US-0, creates the potential for a negative user story, NUS.

US-0. A customer can make a payment using a credit card to purchase a shopping item.

NUS-0. A fraudster wants to steal credit card information to make unauthentic charges.

The initiative to reduce the risk posed by NUS to US-0 motivates (at least) the following interrelated positive user stories: US-1, which is an evolution of US-0, and US-2 and US-3, which are new creations (assuming they did not exist).

US-1. A customer can receive an e-mail message asking him or her to confirm a purchase using a credit card to shop assuredly.

US-2. A customer can create an account on the shopping system to shop serenely.

US-3. A customer can receive an e-mail address as part of his or her account on the shopping system to send messages to and receive messages from the shopping system.

It is customary for a positive user story to be associated with *acceptance criteria*. The acceptance criteria for US-1 needs to consider NUS-0, and, in doing so, must contain appropriate tests to ensure that the purchase is legitimate before processing the payment. For example, one test could check the customer’s e-mail address is indeed in his or her profile and another test could check the contents of customer’s response to the e-mail message.

F. Positive and Negative User Story Relationships

The realization of a positive user story in the software system creates the potential for one or more negative user stories. For example, existence of US-1 creates the potential for a negative user story involving clone phishing (specifically, e-mail spoofing).

The converse is also holds, that is, a negative user story is always associated with one or more positive user stories. For example, if there is no provision for making a payment using a credit card, then there is no need for US-0, and, in turn, no potential for credit card fraud in the given context and, therefore, no possibility of NUS-0.

G. Impact of Negative User Stories on the Positive User Story Relationships

The software requirements, in general, and positive user stories, in particular, can be interrelated in many different ways, such as *Causal*, *Essential* (Is-Dependent-On, Is-Constrained-By), *Implementational* (Conflicts-With, Is-Cost-Related-To, Is-Similar-To, Is-Value-Related-To), *Spatial* (Is-Aggregated-With, Is-Generalized-To, Is-Refined-By), and *Temporal* (Is-Sequential-To, Is-Interleaving-With, Is-Synchronized-With). For example, US-2 is an aggregate of a number of positive user stories, including US-3 and US-4.

US-4. A customer can supply his or her postal address as part of his or her account on the shopping system to receive shopping items.

For software engineers, an in-depth understanding of interrelationships among positive user stories is necessary for a number of reasons, such as prioritizing, scheduling, and release planning the positive user stories properly, testing the positive user stories adequately, and modifying non-independent positive user stories relatively easily.

The existence of a negative user story can influence ways in which positive user stories are interrelated. If a negative user story leads to a change of a positive user story (say, US), then, evidently, the other positive user stories that depend in some way on US can be affected and may need to be changed, too. These changes may not be isolated and can, in fact, *propagate* due to the presence of certain properties (such as symmetry and/or transitivity) of relationships. For example, if there is a set of positive user stories related by the Is-Synchronized-With relationship, then a negative user story that affects one positive user story in that set will affect all positive user stories in that set, as the Is-Synchronized-With relationship is symmetric as well as transitive.

IV. DIRECTIONS FOR FUTURE RESEARCH

A. Other Negative Uses

In recent years, agile methodologies are being applied for the development of airline reservation systems and healthcare information systems [10]. These systems need to be concerned with privacy, safety, and security, and, at the same time, be able to provide a positive UX.

For example, for an airline reservation system, a realization of the following negative user story is (a case of eavesdropping and therefore) a violation of privacy:

NUS-1. A prankster wants to intercept an HTTP cookie to be able to monitor people's travelling habits.

For another example, for a healthcare information system, a realization of the following negative user story is a violation of privacy, safety, and security:

NUS-2. An identity thief wants to steal the medical records of patients to coerce.

Therefore, exploring homogeneous and heterogeneous combinations of violations of privacy, safety, and security, with due consideration for accessibility and usability, is of research interest.

B. Empirical Studies

Traditionally, most agile methodologies do not have native support for extensive conceptual modeling. Therefore, highlighting the human and social challenges inherent in the constructions of the conceptual models is of research interest. Finally, for assessing the feasibility of a deployment of the positive user story engineering process extended by negative user stories, it can be useful to conduct empirical studies in organizations with appropriate agile process maturity levels.

V. CONCLUSION

It is crucial for the organization to cultivate a culture for *proactively* and *cost-effectively* identifying, understanding, and (hopefully) preventing, negative uses of the products and services it provides, so as to sustain confidence of its customers and other stakeholders, to manage its requirements debt, to retain its share and reputation in the market, and to be perceived as socially responsible.

In the context of an agile project, such a commitment requires adequate preparation as early as possible, that is, during conceptual modeling for understanding the problem and positive user story engineering. In particular, if a negative user story with a significant RE level is identified, then, as this paper has attempted to show, it is incumbent upon the development team to take preventative measures.

REFERENCES

- [1] K. Duran, G. Burns and P. Snell, Lehman's Laws in Agile and Non-Agile Projects. The Twentieth Working Conference on Reverse Engineering (WCRE 2013), Koblenz, Germany, October 14-17, 2013.
- [2] W. Kim, O.-R. Jeong, C. Kim and J. So, The Dark Side of the Internet: Attacks, Costs and Responses. *Information Systems*, 36(3): 675-705, 2011.
- [3] J. McDermott and C. Fox, Using Abuse Case Models for Security Requirements Analysis. The Fifteenth Annual Computer Security Applications Conference (ACSAC 1999), Scottsdale, USA, December 6-10, 1999.
- [4] G. Sindre and A. L. Opdahl, Eliciting Security Requirements by Misuse Cases. *Requirements Engineering*, 10(1): 34-44, 2005.
- [5] J. Peeters, Agile Security Requirements Engineering. The Symposium on Requirements Engineering for Information Security (SREIS 2005), Paris, France, August 29, 2005.
- [6] P. Kamthan and N. Shahmir, Modeling Negative User Stories is Risky Business. The Seventeenth IEEE International Symposium on High Assurance Systems Engineering (HASE 2016), Orlando, USA, January 7-9, 2016.
- [7] M. Cohn, *User Stories Applied: For Agile Software Development*. Addison-Wesley, 2004.
- [8] A. Cooper, R. Reimann, D. Cronin and C. Noessel, *About Face: The Essentials of Interaction Design*. John Wiley and Sons, 2014.
- [9] P. G. Armour, The Five Orders of Ignorance. *Communications of the ACM*, 43(10): 17-20, 2000.
- [10] S. A. Fricker, C. Thümmler and A. Gavras, *Requirements Engineering for Digital Health*. Springer International Publishing, 2015.

Quantifying and Assessing the Merge of Cloned Web-Based System: An Exploratory Study

Jadson Santos

Department of Informatics and Applied Mathematics
Federal University of Rio Grande do Norte, UFRN
Natal, Brazil
jadsonjs@gmail.com

Uirá Kulesza

Department of Informatics and Applied Mathematics
Federal University of Rio Grande do Norte, UFRN
Natal, Brazil
uira@dimap.ufrn.br

Abstract— This paper presents an exploratory study that analyzes the complexity to integrate existing merge conflicts of a cloned large-scale web system. The study is supported by an existing tool that focuses on the identification of merge conflicts that can arise during the integration of cloned systems. The approach addresses the merge conflict analysis through the extraction and comparison of the issues and code history of cloned systems using mining software repository and static analysis techniques. The main aims of our study are: (i) to quantify the kind of conflicts defined by our approach that happen when evolving cloned systems; (ii) to evaluate if they are being correctly detected by our tool; and finally (iii) to analyze the difficult to integrate them from one cloned system to another. The study findings show: (i) a predominance of semantic conflicts between issues of source and target cloned systems; and (ii) the feasibility to use merge analysis approaches to integrate tasks from one clone to another.

Clone-and-own approach, web-based information systems, software merge, code merge conflicts

I. INTRODUCTION

Modern software development involves the parallel working of software developers, who work separately on creating and modifying their local copies of code assets, and then need to submit them to version control repositories. In this context, software merge techniques are used to promote the integration of code assets modified in parallel by different developers. When merging code to repositories, conflicts naturally emerge due to code change operations on the same code elements – such as classes or methods, and also because of semantic dependencies between the modified classes [1] [2] [3] [4].

Recent research work [3] [4] [5] has investigated and proposed approaches for the detection and analysis of code merge conflicts. Zimmermann et al [5] analyzed four open source systems from CVS repositories and found a substantial amount of textual merging conflicts – 23% to 43% – in those projects. Brun et al [3] investigated nine active open source projects and found that 16% of the analyzed merge operations contain textual conflicts and a significant number presents high-order conflicts – related to changes semantically

incompatible that causes compilation or test errors. Guimarães & Silva [4] propose an automated approach that continuously analyzes and detects merge conflicts from committed and uncommitted changes with the main aim to anticipate and present them to software developers. Their approach works with a rich set of different merge conflicts.

In addition, Dubinsky et al [6] conducted an exploratory study to investigate the cloning culture in six industrial product lines. They observed that companies usually clone existing products with the main aim of addressing new requirements and customize those products to new contexts and scenarios. In their study, they conclude that cloning is considered a favorable reuse approach that facilitates independent customization of new products based on existing ones. However, they also noticed that the cloning practice brings difficulties when performing maintenance and evolution activities, such as propagating changes between those clones, and integration of the cloned code assets. The authors suggest that clone management techniques could be explored in future research work.

While recent empirical studies have already given a perspective of the different merge conflicts that happen in existing open source systems, there is no empirical study providing a quantitative and qualitative detailed view of how those conflicts are happening and the complexity to integrate them in the context of cloned commercial systems. In addition, the understanding of the complexity of merging cloned systems and the development of techniques to help this activity is also of interest of the software product line community, which has recently identified [6] that cloned techniques are used to manage variabilities from a software product line (or software family).

In this context, this paper presents an exploratory study that quantifies and analyzes the complexity to integrate existing merge conflicts of a cloned large-scale web system. The main aim of our study is: (i) to quantify the three kind of conflicts – structural, semantic and lexical – defined by our approach that happen when evolving cloned systems; (ii) to evaluate if those conflicts are begin correctly detected by our tool; and (iii) to analyze the difficult to integrate them from one cloned system to another. The study is supported by an existing tool [10] that focuses on the identification of merge conflicts that can arise during the integration of cloned systems. The approach

DOI reference number: 10.18293/SEKE2016-232

addresses the merge conflict analysis through the extraction and comparison of the issues and code history of cloned systems [7] using mining software repository and static analysis techniques. The contributions of our study are as follows: (i) it performs a systematic characterization and analysis of the kind and complexity of merge conflicts for large-scale web-based systems; and (ii) it shows the feasibility to use merge analysis approaches to integrate tasks from one clone to another.

The rest of this paper is organized as follows. Section II describes the study settings. Section III presents the study results. Section IV discusses threats and limitation of our study results. Section V reports the related work. Finally, Section VI presents concluding remarks and future work.

II. STUDY SETTINGS

The main aims of our study are: (i) to understand the kind of conflicts that happen when evolving cloned web-based systems; (ii) to evaluate the conflict detection by our tool; and (iii) to analyze the complexity to integrate issues developed for one cloned system – called the source system – to another one – the target system.

A. Categorization of Code Merge Conflicts

In the first step, we considered a categorization of code merge conflicts based on existing research work [2] [3] [4]. A merge conflict is a pair of code changes developed for the source and target cloned systems, which interfere each other when merging code changes from the source to the target. In our study we have focused on the following kinds of conflicts that happen in the context of 3-way merging [1] – which uses information from a common ancestor besides the one available for the two classes being merged:

- i. *direct conflict* (structural) – represents a pair of code changes applied to the same code elements (e.g., attributes, methods) by both source and target systems;
- ii. *indirect conflict* (semantic) – happens when code changes applied to the source system are in the call graph of other code changes in the target system; and
- iii. *pseudo conflict* (lexical) – the source and target systems modify the same class or interface, but different and independent code elements (attribute or method).

We quantify pseudo conflicts only to understand the additional effort of developers when merging cloned systems using textual merge tools. Textual-based tools usually exhibit these conflicts, but the usage of more advanced merge tools (e.g., analysis of direct conflicts) can avoid this additional effort.

B. Data Collection and Analysis Procedure

To support the analysis of conflicts in the merge process, we performed the following data collection procedures, which were automated by a tool we developed:

Step 1: Mine Evolution History. First, the MergeClear tool mines the development issues (change requests) of the cloned

web systems from issue tracking systems (such as Bugzilla) used by each institution. It produces as output the task evolution history file, an XML file (one for the source and another one for the target system) that contains all the development issues (with information such as description, version, kind of issue, related modules, etc), and respective code revisions associated to them.

Step 2: Mine Code Evolution. After that, the tool recovers and compares subsequent revisions associated to the Java code assets mined from version control systems to extract fine-grained code change operations applied to them. In particular, in this study, we focused on the following code change operations: addition, deletion and modification of classes, attributes and methods. Our tool can currently also quantify code change related to class inheritance, interface implementation, and code annotations, but they were not explored in our study. All this information is stored in the change log history file, which is a refinement of the issue evolution history.

Step 3: Analysis of Merge Conflicts. Next, the tool processes and compares these change log files produced for the source and target cloned web-based systems to automatically calculate the direct, indirect, and pseudo conflicts between them. The direct conflicts are quantified by identifying attributes, methods and classes that were modified in both source and target systems. The computation of indirect conflicts also utilizes the system call graph to quantify which of the code change operations in the source system are in the call graph of other code change operations from the target system. Pseudo conflicts are calculated by identifying code change operations applied to the same classes and do not have indirect conflicts. The tool can also be used to visualize the list of development issues and associated code changes applied to the source and target web-based systems. In addition, we also present the list of code conflicts that will occur when merging the development issues of the cloned web-based systems with their respective conflicting code changes.

Step 4: Group and Order Issues. After the conflict analysis step, the issues were ordered and grouped to represent different integration scenarios. The main criteria were (i) the kind of the issues (ii) the kind of conflicts detected in the issue; (iii) the amount of source code artifacts changed in the issues; and (iv) if the issues change different layers or modules of the system or if they only modify restricted code.

Step 5: Issues Selection. After the issue classification, a specific set of issues was selected to represent different integration efforts. In our study, we have not analyzed issues with pseudo conflicts because they do not represent real conflicts when merging code from source to the target system [12]. For all the selected issues, we used the merge functionality of the subversion plugin to integrate code changes from the source to the target system and compared with the results indicated by our tool. Fig. 1 illustrates this analysis workflow.

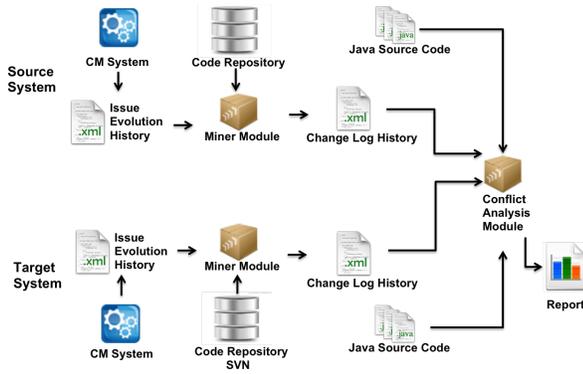


Figure 1: Automated Approach to Reconciliation of Cloned Systems

After the tool execution, we conducted a study guided by the following research questions:

- RQ1. Which amount and kind of code merge conflicts happen when evolving and merging cloned web-based systems?
- RQ2. Do issues without merge conflicts can really be automatically integrated?
- RQ3. Are direct conflicts correctly identified? How complex they are to be integrated?
- RQ4. Are indirect conflicts correctly identified? Do the indirect conflicts can cause behavior problems on the target system as the approach suggest?

C. Target Web-based System

The Federal University of Rio Grande do Norte (UFRN, Brazil) develops a set of web information systems designed to automate business processes for universities focusing on different and complementary aspects, such as academic, administration, planning and management. These systems began to be deployed in 2006. Our study focuses on one of these systems. Table I shows an overview of the system in terms of users and size.

TABLE I. SIZE AND USERS OF ANALYZED SYSTEM

System	Total of Users	Daily Access	KLOC	Java Classes	Java Methods
SIGAA	52000	56000	833	5906	81102

The SIGAA system was chosen because it is a large-scale web-based information system implemented in the Java language. The system uses the clone-and-own strategy to promote its customization to other partners. Also, we have complete access to the version control and issue tracker systems of the development teams, responsible for the source and target systems. The scenario chosen for this study involved a total of 1083 issues.

III. STUDY RESULTS

This section presents and discusses the obtained results for our research questions.

A. RQ1: Which amount and kind of code conflicts happen when evolving and merging cloned web-based information systems?

We analyzed the amount and kinds of conflicts considering the perspectives of development issues and code change operations. Table II displays the distribution of conflicts detected by our tool in the analyzed issues. 65,18% of issues presents no conflict. 5,6% of issues had only direct conflicts. 18,83% of issues held no direct conflicts but present at least one indirect conflict, and 10,34% of issues have only pseudo conflicts.

Table III shows the amount of code conflicts that were calculated for the merge of the investigated cloned web systems. Most of the collected code conflicts (68,14%) are indirect. It means that a great number of semantic conflicts that happen when merging web information systems are not made explicit when only using textual-based merging tools. Developers should be aware of them to understand the change impact of the merging, and to identify specific parts of the system – the affected call graphs – to be re-tested after the integration. Our study also found a significant number of pseudo conflicts (22.20%). They represent additional useless effort from the developers when merging classes with existing textual-based tools, because those tools are not able to identify independent code changes applied to the same classes. Finally, Table III shows that a reduced number of direct conflicts (9,65%) were found in our study compared to the other kinds. Those direct conflicts require the developer intervention to be merged from the source to the target web-based system, because they represent direct changes applied to the same code elements (attributes, methods).

TABLE II. AMOUNT OF ISSUES BY KIND OF CONFLICT

	Without Conflicts	With Direct Conflict	With Indirect Conflict	With Pseudo Conflict
Amount of Issues	706 (65,18%)	61 (5,6%)	204 (18,83%)	112 (10,34%)

TABLE III. KINDS OF CONFLICTS DETECTED

Kind of Conflicts	Direct Conflict	Indirect Conflict	Pseudo Conflict
SIGAA	120 (9.65%)	847 (68.14%)	276 (22.20%)

B. RQ2: Do issues without merge conflicts can really be automatically integrated?

This research question focuses on evaluating if issues that were classified as having no conflicts by our approach can be automatically integrated using the traditional merge mechanisms from existing control version systems.

In order to answer this research question, we have selected a total of 15 issues between those ones identified as having no merge conflicts by our approach (Table II). After that, we identified the different commits and respective code changes associated with each issue, and they were manually applied from source to the target cloned web system using the merge

functionality of the subversion plugin. Finally, we verified for each issue if there was no compilation error after the code merge of the different commits associated to the issue.

Table IV shows the results of our analysis. We have found that 11 issues of the source cloned web system from the 15 analyzed (73%) could be integrated without causing any compilation errors in the target web system.

TABLE IV. ANALYSIS RESULT OF THE RQ2

Analyzed Issues	Possible to Integrate	Dependence Error	Mining Error	No Manual Merge Possible
15	11 (73.33%)	1 (6.6%)	1 (6.6%)	2 (13.33%)

During our analysis we have observed that one issue exhibited a dependence error, which indicated that this issue could only be integrated if another one is integrated previously. It means that only the criterion of conflicts between code artifacts was not enough to ensure that the integration occurred without adding compilation errors in the target system. Because of that, it was also necessary to analyze the dependencies between issues. Thus, an issue from the source system could be integrated without compilation errors in the target system only if it has no conflicts and all the issues that it depends on were integrated before it.

The dependency computation algorithm used in our study is based on the data of the issue creation. This criterion was not enough to determine whether an issue has been implemented before another one or not. In some cases it is not trivial to determine whether an issue was accomplished before another one, because they occur in parallel, having interleaved commits. This has generated a mining error during the integration of one issue (Table IV) and it needs to be improved in the conflict analysis module.

Finally, there are two issues from the source system, which are not possible to merge in the target web system (Table IV). In one issue, some classes have been changed, but part of this evolution was registered in another issue. Thus, it brought difficulties to manually separate the source code belonged to the specific issue before proceed with the automatic merge. The second issue received the first commit in 2010, but the issue was not totally finished. Only in 2012, the issue was completed. Meanwhile, several commits were registered which originated many code changes for other issues, thus making very complex to perform a manual analysis of the issue, and separating just its specific changes. The discovering of such cases revealed the need to correctly register the association between issues and respective commits in the issue tracker or control version systems.

C. RQ3: Are direct conflicts correctly identified? How complex they are to be integrated?

In this research question, we have analyzed the direct conflicts identified by MergeClear tool in order to determine whether they were correctly identified and how complex they are to be integrated from the source to the target system. In short, we have analyzed and compared the code evolution of the source and target systems to confirm the existence of the

identified direct conflicts and verify if those code changes can still coexist in the target system.

To answer this research question, we have selected and manually analyzed 10 different issues. All the direct conflicts identified by MergeClear tool represent real conflicts, so we did not find any false positive. In addition to that, we also analyzed the complexity to integrate the issues with such direct conflicts. Table V shows the results for this analysis. As we can see, 60% (6 from 10) of the issues that exhibited direct conflicts could be integrated. It means that although there were code changes applied to the same artifact (method or field) in the source and target systems, they have been applied to separate parts of the code, and they are not directly related or incompatible. However, due to the difficult to automatically analyze the different semantic of such code changes, it is always necessary a manual analysis to verify the possibility of integration. Fig. 2 shows an example of a change that was considered possible to be integrated. In a same method the changes of the source and target systems were accomplished in separated “cases” of a Java switch statement. These changes are not strongly related, and they can coexist.

On the other hand, our analysis also revealed that 40% of the investigated issues (4 of 10) exhibit complex direct conflicts, which are difficult to be integrated even when applying a manual merge. Fig. 3 illustrates a change that was accomplished in the same functionality of a specific method of a class for both the source and target systems. Because these changes involve overlapping updates to the implementation of the same functionality, they are difficult to merge.

TABLE V. ANALYSIS RESULT OF THE RQ3

Analyzed Issues	Direct Conflict Possible to be Integrated	Direct Conflict Improbable to be Integrated
10	6 (60%)	4 (40%)

```

case StatusOperation.REVERT_STUDENT_EXIT:
    student.setStatus(Student.ACTIVE);
    break;
change in source

case StatusOperation.FINISH_STUDENT:
    if(student.isSpecial())
        addErrorMessage("it is not possible");
    break;
change in target
    
```

Figure 2: Direct Conflict without Integration Problem

```

public class ClassX {
    public void doBusinessRuleX(){
        if(doStuff()){
            // do something
        }else{
            sendMail();
        }
    }
}
change in source

public class ClassX {
    public void doBusinessRuleX(){
        if(doStuff() && anotherThing()){
            // do something
        }
    }
}
change in target
    
```

Figure 3: Direct Conflict with Integration Problem

D RQ4: Are indirect conflicts correctly identified? Do the indirect conflicts can cause behavior problems on the target system as the approach suggest?

The aim of this research question was to analyze whether the indirect conflicts collected by the MergeClear tool were correctly identified and if they could affect the behavior of the merged functionality of the changes developed for the source and target systems. In other words, we have investigated if the integration of the source and target changes that are related to indirect conflicts can cause any abnormal behavior in the system or not.

The process of manual analysis for this research question is similar to the direct conflict analysis, differing that the comparison of the evolution in the source code was made between artifacts (methods or fields) in a certain level of the call graph, and verifying if the application of this change add some error to the target system.

Table VI shows the amount of indirect conflicts identified for the evolution of the clones of the web system analyzed in this study. They were organized by the level in the call graph where they were detected. For this study we have focused on the analysis of the depth level maximum of 3. As you can see, most of indirect conflicts detected by our tool are at level 1 and 2, which justifies the analysis of indirect conflicts until the level 3.

TABLE VI. INDIRECT CONFLICTS BY LEVEL

Level	Number of Indirect Conflicts	Percentage
1	338	38.72%
2	327	38.61%
3	192	22.67%

To answer RQ4, we have manually analyzed a total of 12 different issues that contains indirect conflicts. All the 12 issues analyzed represent indirect conflict defined in our approach. After that, we analyzed if those indirect conflicts could be integrated without presenting any error. We have found that 7 issues (58%) actually could add behavior problems after the integration. The analysis of the 5 remaining issues (42%) showed us that they could be integrated without causing behavior problems to the target system. This result reinforces the need to re-test issues that involves the existence of indirect merge conflicts. Table VII summarizes such results.

TABLE VII. ANALYSIS RESULT OF RQ4

Analyzed Issues	Indirect Conflict can not add behavior problem to target system	Indirect Conflict can add behavior problem to target system
12 (100%)	5 (41.66%)	7 (58.33%)

Fig. 4 shows an example of changes from one analyzed issue that generated indirect conflict in the call graph but it did not cause any behavior problem after the merge process. This example represents an extract method refactoring, which was

only applied to improve the maintainability of the code. Those kinds of change in the source web system although have exhibited indirect conflicts with other code changes in the target system, they could be automatically merged without presenting any behavior problem to the target system.

```

public void createAutomaticSpreadSheet(StudentClass _class){
    if(!_class.isOpen()){
        populateData(_class, student);
    }

    try{
        executeCreateClass(_class);
    }catch(Exception ex){
        addErrorMessage(ex.getMessage());
    }
}

private void populateData(StudentClass _class, Student student) {
    _class.addStudent(student);
    _class.setActive(true);
    // ..
}
    
```

Figure 4: Indirect Conflict not affect target system behavior

Fig. 5 represents a change in the target system responsible to perform a specific validation for a certain kind of student. However, the original generic validation in the source system was changed during the parallel evolution of the cloned systems. Because the change in the source system did not consider the specific validation codified in target system, the merge of this code can add behavior problem to the target system.

```

if(isClosedClass()){
    ClassValidator.validate(_class);
}

if(isClosedClass()){
    if(!_class.isUndergraduate()){
        NewClassValidator.validate(_class);
    }else{
        ClassValidator.validate(_class);
    }
}

public class ClassValidator {
    public static void validate (StudentClass _class) {
        // business rules omitted
    }
}
    
```

Figure 5: Indirect Conflict affect the target system behavior

IV. THREATS TO VALIDITY

Our study has focused only on the analysis of a restricted set of merge conflicts and code change operations. Other kinds of merge conflicts – such as language, semantic and test conflicts [4] – are planned to be included in our mining tool for future studies. Regarding the code change operations, our tool is currently been extended to also analyze changes on class inheritance, interface implementation, and code annotations. The results of our study are restricted to the context of the investigated web-based system, and cannot be generalized. The selection of just one clone of source and target system evolutions also represents a threat, although the selected clone contained thousands of issues. In order to expand our results, we need to replicate it for other existing systems and domains. In this direction, we plan to conduct replications of our study in the context of other existing clones from the same web-based systems presented in this paper, and to existing open-source systems from GitHub repository.

V. RELATED WORK

Recent research work [3] [4] has investigated and proposed approaches for the analysis of code merge conflicts. Guimarães & Silva [4] propose an approach for the early and continuous detection of merging conflicts from uncommitted and committed changes in order to anticipate problems and to avoid overloading developers. They conduct an empirical study that brings evidence that the approach contributes to improve the early detection of conflicts and to avoid overloading developers in comparison with existing approaches. Brun et al [3] also conducted an empirical study considering two kinds of conflicts: (i) textual conflicts – that represent conflicts from code changes in the same artifacts; and (ii) high-order conflicts – that represent code changes that do not generate textual conflicts, but on the other hand, they cause semantic problems – compilation or test failures. Their study found for nine open-source systems that 16% of all merges present textual conflicts, and 33% of merges with no textual conflicts contain high-order conflicts. They present a quantitative and preliminary approach evaluation. In contrast, our work conducted an exploratory study of clones of a large industrial web system that quantified existing merge code conflicts and conducted a detailed analysis on the accuracy and complexity of integrating those merge conflicts using automated support.

Apel et al [12] have argued that a significant number of conflicts are ordering conflicts and show that the usage of semi-structured merge can reduce conflicts when compared to unstructured merges. Our work is consistent with their results, given the percentage of indirect and pseudo conflicts observed. However, ours is an exploratory study to better characterize and understand the kind and complexity of merge conflicts that happens in a large-scale web-based system.

Dubinsky et al [6] conducted an exploratory study of cloning in six industrial software product lines (SPLs). They found that cloning SPLs is considered a reuse approach that facilitates the independent customization of new products based on existing ones, although on the other hand, it can bring difficulties to perform maintenance and evolution activities. The merge conflict analysis approach developed presented in this paper can be used to automatically identify and possibly promote the integration of issues from one SPL clone to another. In addition to that, our work has presented concrete data related to the integration of existing cloned large-scale web systems.

Rubin et al [7] [8] propose a framework for organizing knowledge related to the development, maintenance and merge-refactoring of product lines realized via cloning. They organize such framework in terms of a set of clone management operators. They also performed a detailed analysis of development issues of industrial SPL companies in terms of these operators. Indeed, the web-based system investigated in our work can be seen as a cloned product lines that is evolving independently to accommodate new variabilities. In this paper, we have proposed a merge conflict analysis approach to understand and promote the integration of

development issues that can also be used in the context of cloned SPLs. While Rubin et al [7] [8] propose a general and language independence approach, we restrict our approach to system implemented in the Java language, which allowed achieve more concrete results in our analysis.

VI. CONCLUSION

This paper presented an exploratory study of characterization of merge conflicts in the context of cloned web-based information systems. In our study, we have found: (i) a considerable number of indirect merge conflicts compared to direct and pseudo conflicts when merging cloned web-based-systems; and (ii) the feasibility to use merge analysis approaches to integrate tasks from one cloned system to another one considering the kinds of merge conflicts analyzed in our study – direct, indirect and pseudo. Finally, we also found that the integration of issues from source to target systems also requires the computation and resolution of dependent issues that were previously developed in the source system. As a future work, we plan to replicate our study to other cloned web-based information systems from our institution and from other companies in order to have a better understanding of the cloning impact for this domain of applications. In these new studies, we are also including other categories of conflicts and code change operations. It is also necessary to improve the dependency detection algorithm to address the integration of dependent issues.

REFERENCES

- [1] T. Mens. A State-of-the-Art Survey on Software Merging. *IEEE Transactions on Software Engineering (TSE 2002)*, Vol. 28, 5.
- [2] J. Wokla, B. Ryder, F. Tip, X. Ren. Safe-Commit Analysis to Facilitate Team Software Development. *Proceedings of ICSE 2009*.
- [3] Y. Brun, et al. Early Detection of Collaboration Conflicts and Risks, *IEEE Trans. on Software Engineering*, vol. 39, no. 10, pp. 1358-1375.
- [4] Guimarães, M. L. & Silva, A. R. 2012. Improving early detection of software merge conflicts. In *Proceedings of the 2012 International Conference on Software Engineering (ICSE 2012)*. IEEE Press, Piscataway, NJ, USA, 342-35
- [5] T. Zimmermann, "Mining Workspace Updates in CVS," *Proc. Fourth Int'l Workshop Mining Software Repositories*, May 2007.
- [6] Y. Dubinsky, et al. An Exploratory Study of Cloning in Industrial Software Product Lines. *Proceedings of CSMR 2013*.
- [7] J. Rubin, et al. 2012. Managing Forked Product Variants. *Proceedings of the Software Product Line Conference (SPLC 2012)*. Salvador. Brazil
- [8] J. Rubin, K. Czarnecki, M. Chechik. Managing cloned variants: a framework and experience. *Proceedings of SPLC 2013*: 101-110.
- [9] G. Lima, et al. A Delta Oriented Approach to the Evolution and Reconciliation of Enterprise Software Products Lines. *ICEIS (1) 2013*: 255-263
- [10] MergeClear. A tool for merge cloned systems <http://github.com/jadsonjs/MergeClear> (last visited on August 27, 2015)
- [11] Product Line Hall of Fame. <http://splc.net/fame.html> (last visited on August 27, 2015)
- [12] S. Apel, J. Liebig, B. Brandl, C. Lengauer, C. Kästner: Semistructured merge: rethinking merge in revision control systems. *Proceedings of FSE 2011*
- [13] MergeClear Wiki. Welcome to the MergeClear wiki. <http://github.com/jadsonjs/MergeClear/wiki> (last visited on August 27, 2015)

Automaticly Generating Web Page From A Mockup

Ruozi Huang

School of Data and Computer
Science, Sun Yat-san University
Guangzhou China
huangrz@mail2.sysu.edu.cn

Yonghao Long

School of Data and Computer
Science, National Engineering
Research Center of Digital Life,
Sun Yat-san University
Guangzhou China
longyh3@mail2.sysu.edu.cn

Xiangping Chen*

Institute of Advanced
Technology, Sun Yat-san
University, Guangzhou China
Research Institute of Sun Yat-sen
University in Shenzhen, China
chenxp8@mail.sysu.edu.cn

Abstract—UI is an important part of software product. Considering the complexity of web UI, generating the web page from a mockup proposes requirements for rich experience of developer. Extracting visible elements and their relationship, selecting proper tags, generating source code are time-consuming and error-prone task. In this paper, we propose a method to automate the transforming of the mockup to the web page. Our approach starts from the mockup designed by the art designers, and extracts the elements based on the color features of the edges. Then a bottom-up tag generating method based on the Random Forest is proposed to select the tags for elements. Finally the web page is generated by the definition of the elements. The generating tags can achieve an average accuracy of more than 84%, which can meet the basic requirements of the developers.

Keywords—web generating, machine learning, web design

I. INTRODUCTION

As an important part of the software products, the user interface (UI) builds a bridge between the end-user and the functionality. Well-designed UI have good usability and aesthetics, which will attract the users. However, getting a satisfying UI is not easy. The coding includes understanding complex widgets, trying different prototypes to achieve good user experience, and a number of layout strategies. These requirements limit the speed of UI development.

Generally, the UI of web page is relatively complex. Firstly, the number of element is larger. The number of DOM nodes in most web pages is between 50 and 200 [1], while the number of elements in UI of most app is between 10 and 30 based on the statistics of 61,089 UI pages[18]. Secondly, there are numerous tags to be used to define UI elements. For some tags with similar usage context, choosing tag becomes difficult for non-expert developers.

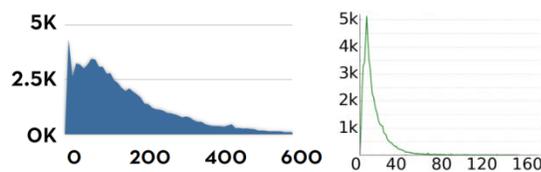


Figure 1. Number of nodes in a web page and UI page.

Many works [8-13] had been proposed to simplify or automatic the UI development process. Modern development environments provide some tools to support the UI development. However, choosing the proper widgets is confusing or non-expert developers. Example based approaches [8, 9] are proposed considering large amount of UI examples in the internet. Examples can tell the developers how to arrange the layout and use the proper widgets. These approaches show that knowledge in the examples is useful to be reused in generating color scheme, device-adaptive UI, and the layouts. However, no approach is proposed for guiding tag selection in web page implementation.

Typical process of developing a draft web page includes the following steps: 1) a wireframe is designed to show the basic functions and structures of the page; and 2) the art designers design a mockup based on the wireframe, the mockup can be regarded as a screenshot of the web page with well designed layout and color theme; 3) the developers extract the elements from the mockup and select tags for the elements; finally 4) developers generate the hierarchical structure of the elements and write HTML and CSS codes to generate a prototype of the web page.

We notice that the process of transforming the mockup to the prototype of web page has the potential to be automated. The edges of rectangles recognized from mockup can be used to extract elements and their relationship. In addition, uniform and standard usage of most tags, which can be learnt from the existing webs. It will save time for developers if they can get a prototype quickly and focus more on the page refinement.

In this paper, we propose a method to generate a draft web page from the mockup. An algorithm is proposed to extract the elements from the mockup based on the wireframe of the mockup. According to the nested relations of the elements, we can get a hierarchical tree of the elements' structures. Since most tags have uniform using in existing web pages, we propose a bottom-up tag generating algorithm to choose tags for each element. This algorithm is based on the Random Forest method. Finally, a prototype of web page is generated by the definition of these elements.

We choose 50 web pages from different websites to verify our method. In the experiment, our algorithm selects the tags

for 9627 elements with the accuracy of 84.4%. The generated web page can meet basic requirements and provide as a prototype.

The remainder of the paper is organized as follows: Section 2 discusses previous work which is related to our work; Section 3 introduces how to generate the web’s source code from a mockup; Section 4 presents the experiments for evaluating the accuracy of the generating tag to the actual tag designed by developers. Finally, Section 5 concludes our work.

II. RELATED WORK

Generating a satisfying Graphical User Interface is very important but difficult in the software development. Most interfaces are still hand-coded, even though there have been a great number of tools to simplify or even automatic the process of GUI generation. Brad et.al [2] had summarized the state of GUI design tools in 2000.

A. User Interface Generation

A number of modern development environments have proposed tools to support the GUI development such as the Dreamweaver, Eclipse, and Visual Studio etc. Developers can drag and drop widgets into containers and set the various properties of the widgets, the basic interface code will be generated once the UI is designed. These tools are very convenient for generating a prototype of the UI but are not ideal in that a) the code that is generated may not be in a style or form the programmer desires; b) once the code is modified it becomes difficult to use the support to update or change the user interface; c) the generated code often use absolute positions and it is complex to generate easily resizable interfaces and d) it’s hard to choose a proper widget for the beginners especially when they are dealing with numerous kinds of widgets.

Another way of getting UI software is searching the existing software from the internet. Developers can get the UI by inputting some keywords in some websites such as Github (www.github.com), Krugle(www.krugle.org), and Open Hub(www.openhub.net) etc. These websites provide rich UI sources but getting a corresponding UI which is close to the developer’s requirement is not easy. A suitable searching result needs a set of accurate keywords and a long time searching from the resulting candidates. Some works focused on the improvement of searching engine [3, 4] and simplifying the process of searching [5, 6], which let the UI searching more convenient and convincible. Developers can use the existing UI for inspiration or studying, they can also reuse the UI but generating an original UI is still a tough work.

B. GUI Redesign

The user interface design often involves the rapid iterative design, exploration and comparison of different interface implementations [7]. Lots of change will occur during the development of UI like the change of color theme, widget sizes and positions. It takes a lot of energy in changing the user interfaces that some works try to automate the refactoring or redesigning process. Kumar et.al [8, 9] proposed an example-based webpage retargeting method, which enables developers choose an existing web as the example and change the source

page to be the example-liked one. Some works focused on the automatic color redesigning of the web for different requirements like the energy saving [10, 11], adaptive to color-deficiency users [12], and color theme modification [13].

With the development of different display devices, some usability problems occur when the traditional web is displaying on these devices. A number of works were proposed to make the traditional web adaptive to the devices’ sizes. The key problem includes segmenting the existing UI into some semantic parts [14], resize and rearrange the widgets [15], and refactoring the existing source code [16]. A good summary of the adaptive model-driven UI development is provided by [17].

III. GENERATING THE SOURCE CODE FROM MOCKUP

In this section, we describe the method of extracting the elements from a mockup based on the color feature and the shape of the elements. A bottom-up tag generating algorithm is proposed to generate tags for the elements extracted from the mockup. Based on the information on the elements and their tags, we generate the prototype of the web page.

A. Extracting the elements from mockup

The mockup can be regarded as the screenshot of the web page. Noticing that the elements are always rectangle, developers can get the position and size of the element by detecting the edge of the element and extract the sizes and positions. The extracting of the elements in the mockup is finding the separator line in the mockup and extracting the divided blocks. A line is the separator line if it meets the following conditions: 1) the line contains only one kind of color and 2) more than one neighbor line contains different colors. The neighbor line is defined by the following formula:

$$\begin{cases} l_0 : \{y = y_0, x \in [x_0, x_1]\} \\ neighbor(l_0) : \{y = y_0 \pm 1, x \in [x_0, x_1]\} \end{cases}$$

We then proposed the separator generation algorithm which finds the separator line in the mockup: for the current image, we find all the horizontal and vertical separator line by the above definition. These lines separate the image into a set of sub-images. Then we do the same procedure for each sub-image. The algorithm ends when no separator line can be found in the image.

Algorithm: Segmenting(G_i , lines)

Input: An image G_o of visual design.

Output: An image G_o ’with separation lines.

The boolean function isPure(l_m) returns true if the line l_m contains only one kind of color.

Start

if($i=0$)

then add the boards of G_i in lines

else {

for($m=0$ to $G_i.width$)

```

get vertical line  $l_m$  ( $x=m, y \in [0, G_i.width]$ )
if (isPure( $l_m$ ) and (!isPure( $l_{m-1}$ )||!isPure( $l_{m+1}$ )))
    then  $v\_l.add(l_m)$ ;
     $sp\_Line.add(v\_l)$ ;
for( $n=0$  to  $G_i.height$ )
    get horizontal line  $l_n$  ( $y=n, x \in [0, G_i.height]$ )
    if (isPure( $l_n$ ) and (!isPure( $l_{n-1}$ )||!isPure( $l_{n+1}$ )))
        then  $h\_l.add(l_n)$ ;
         $sp\_Line.add(h\_l)$ ;
 $G_0' = drawLine(G_0, sp\_Line)$ ;
 $subG = getSubGraph(G_i, v\_l, h\_l)$ ;
if( $subG.size \leq 1$ ) {
    return;
}
else {
    for( $G_{i+k} \in subG$ )
        segmenting( $G_{i+k}$ );
}
End

```

B. Generating tags for the elements

Choosing appropriate tags for elements is very important in front-end development. Well-designed web pages obey the rules of W3C, which will be friendly with the search engine and easy to be understood by the developers. After observing great amounts of web pages we found that elements with different semantics have differences in the performance of vision and structure, thus the appropriate tags can be gotten by considering the elements' visual and structural properties.

Most tags are used in particular situations; we use the Random Forest method to learn the using of tags. An element's tag is defined both by the predicting result and some heuristic rules proposed by us. The following of this section describes the features we get from the elements for training and a bottom-up tag generating method with some heuristic rules.

1) Extracting the elements' features.

The using of the tag is concerned with the content of the element and its function. Which means an element's tag can be speculated by analyzing the element's visual and structure properties. Thus we extract the elements' visual and structure features, and use the Random Forest method to learn the features of tags' using to predict the elements' tags.

The features we extracted from an element include two kinds. One is the visual properties like the element's sizes, positions, main colors and so on. The other is the structural

properties like the element's level number in the hierarchical tree, its siblings' and children's number, and its neighbors and so on. Detailed description is proposed in Table 1.

2) Generating the appropriate tags for elements

The web contains numerous kinds of tags, some tags like the <DIV> and don't have the exact semantic meaning. Their using is not regular that their visual properties are not particular. But they often appear in the inner nodes and can be inferred by their children. This phenomenon let us do the tag's generating from the leaf nodes first.

In order to know the frequency tags used in the web, we firstly extract 50 web pages tags and count the tags in them, the results are showed in Fig.1. The statistics of the frequency tags shows that the leaf nodes contains many simpler tags than the inner nodes such as <a>, . These tags are often used in a uniform way and they have more outstanding features such as the often contains complex colors, the <a> and often have a big aspect ratio cause they are always a single line of texts, and so on.

As for the tags of inner elements, the using of tags is tending to represent the structure of web, such as the widely used of <div>. Some of these tags can hardly find some clearly visual features: their position and size are diverse. But some structure features are obvious: e.g. a always contains some tags, a <p> will contain some <a> or . But the structure features can be extracted only if the tags of children are determined. Thus we give tags for the leaf nodes at first. This is based on two concerns: 1) leaf nodes have simpler tags and their features are more significant than the inner nodes as Fig. 2(a) shows below, and 2) the determination of the leaf nodes' tags can give a hint to the chosen of their parents' tags, thus the whole elements' tags can be determined recursively.

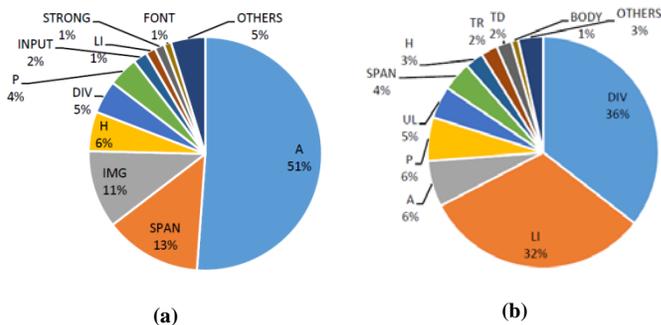


Figure 2. The Top 10th frequency tags used in (a) leaf nodes (A 51%, SPAN 13%, IMG 11%, H 6%, etc) and (b) inner nodes (DIV 36%, UL 32%, A 6%, P 6%, etc).

We use the random forest method to generate tags for the leaf nodes. Since the using of tags of inner nodes is more complex than the leaf nodes, the same method performs badly in this situation. But the leaf nodes' tags can give a hint for finding appropriate tags for the inner nodes. A bottom-up tag generating method is proposed for finding tags for the inner nodes: a node's tags can be determined by its own properties and its children's properties.

TABLE I. DIMENSIONS FOR THE ELEMENTS' TAG GENERATING

ID	Name	Description
1	h	The element's height
2	w	The element's width
3	x	The element's left-top corner's x-value
4	y	The element's left-top corner's y-value
5	r_w	The result of element's width relative to its parent
6	r_h	The result of element's height relative to its parent
7	r_x	The result of element's x-value relative to its parent
8	r_y	The result of element's y-value relative to its parent
9	aspectRatio	width/height
10	Area	width*height
11	Level	Tree level number in the hierarchical tree
12	siblingNum	Number of siblings
13	siblingNum_x	Number of siblings with same x-value
14	siblingNum_y	Number of siblings with same y-value
15	siblingNum_w	Number of siblings with same w-value
16	siblingNum_h	Number of siblings with same h-value
17	colorNum	Number of element's main colors
18	childrenNum	Number of children
19	childrenNum_x	Number of children with same x
20	childrenNum_y	Number of children with same y
21	childrenNum_w	Number of children with same w
22	childrenNum_h	Number of children with same h
23	subtree_height	Height of the element's subtree
24	coverage	Coverage ratio of the element
25	children_type	A string of element's children's types
26	DIV number	Number of children with tag <DIV>
27	P number	Number of children with tag <P>
28	LI number	Number of children with tag
29	UL number	Number of children with tag
30	H number	Number of children with tag <H>
31	FORM number	Number of children with tag <FORM>
32	IMG number	Number of children with tag
33	INPUT number	Number of children with tag <INPUT>

3) Proofreading

The generating tag from the previous section may contain some mistakes; we propose some heuristic methods to proofread the result.

Our proofreading includes finding the missing elements which belong to a list or table; generating the headers and footers according to the elements' positions; and changing the element's tag according to its children.

The elements which can form a list or table have some obvious features. They are arranged in a line and their widths or heights are close, the neighbors are close. The table can be regarded as a set of adjacent lists with same number of elements. Based on the above features, we propose a heuristic algorithm to find the missing elements which belong to list or table.

Algorithm: findList(e)

Input: An element e of the web page.

Output: A tag of the element e.

Start

$E_x = \{e_i \in e.sibling \mid e_i.x = e.x\};$

$E_y = \{e_j \in e.sibling \mid e_j.y = e.y\};$

Table = \emptyset ;

if($E_x.length > 0$ and $E_y.length > 0$) {

// This means they may be in a table

for all $e_i \in E_x$

for all $e_j \in E_y$

if there exists an element e_{ij} s.t.:

$e_{ij}.y = e_i.y; e_{ij}.w = e_i.w; e_{ij}.x = e_j.x; e_{ij}.h = e_j.h;$

then

Table.add(e_{ij});

Table.add(e_i) if it doesn't contain it;

Table.add(e_j) if it doesn't contain it;

for all $e_i \in$ Table {

$e_i.tag = <TD>$

$e_i.parent.tag = <TABLE>$

}

$e.tag = <TD>;$

}

else {

if($E_x.length > 0$) {

$e.tag = ;$

$e_i.tag = (e_i \in E_x)$

if($e.sibling - E_x = \emptyset$) **then** $e.parent = ;$

else generate a new parent element of e and E_x ;

} **else if**($E_y.length > 0$) {

$e.tag = ;$

$e_j.tag = (e_j \in E_y)$

if($e.sibling - E_y = \emptyset$) **then** $e.parent = ;$

else generate a new parent element of e and E_y ;

} **else** {

$e.tag = <DIV>;$

}

}

End

C. Layout generation

From the generating elements above, we can write the HTML source code using a stack to achieve the goal. And a

CSS selector is generated to define the element's visual properties. Fig 3. shows the HTML and CSS code of element *e*.

```
HTML:
<e.tag id="e.tag_e.id">
...
</e.tag>

CSS:
#e.tag_e.id{
width: e.width;
height: e.height;
offsetLeft: e.x-e.parent.x;
offsetTop: e.y-e.parent.y;
background-color: e.getMainColor;
}
```

Figure 3. The HTML and CSS code of element *e*.

IV. EXPERIMENT

A. Implementation of the experiment

The key problem in the web generating from the mockup is guarantee the accuracy of elements' tags. We select 50 web pages from different websites to verify our result. We choose 50 universities' web pages which are following the standards of W3C to do the experiment. Since the elements extracted by the DOM tree are different from those we extracted from the mockup, we do some preprocessing to remove the elements which are fully covered by their children or are not appeared on the web page.

For each time, we use 40 web pages as the training set to predict the left 10 pages. For the generating result, the *prediction* of tag T is the total number of those tags whose tags are predicted to be T. The *ground truth* is the total number of elements whose tags are T. Then we define the *overlap* as the intersection of *prediction* and *ground truth*. And the *precision*, *recall*, and *F1 value* is defined by the following formulas.

$$\begin{cases} precision = \frac{overlap}{prediction} \\ recall = \frac{overlap}{groundTruth} \\ F1 = \frac{2 * precision * recall}{precision + recall} \end{cases}$$

In each time, we record the *precision*, *recall*, and *F1 value* of the generated tags. Then we manually check the result since some tags can be replaced by other tags, the corrected is marked as the accuracy. We will repeat this process until 50 web pages are checked.

B. The analysing of the accuracy of the generating tags

Table 2. records the average accuracy of the generating tags from 50 pages. Since the tag in the leaf nodes

are often mixed with the tag <A>, we divide the two tags into different kinds when they are inner nodes and mix them into one kind if they are leaves. The tag set 'H' means the tag of <H1> to <H6>.

From the result we can see the accuracy of the tag in inner nodes is very low. This is because the number of this tag in 50 pages is very small, the use of this tag can be replaced by the tag <A> in most cases.

TABLE II. THE ACCURACY OF THE GENERATING TAGS

Tag Set	Precision	Recall	F1 Value	Accuracy
DIV	0.609	0.822	0.7	0.651
IMG	0.836	0.782	0.808	0.844
INPUT	0.738	0.674	0.705	0.762
P	0.683	0.499	0.577	0.807
SPAN (inner nodes)	0.5	0.074	0.128	0.964
H	0.619	0.609	0.614	0.723
LI/TH/TD	0.861	0.659	0.747	0.958
UL/OL/TABLE	0.662	0.634	0.647	0.946
FORM	0.419	0.818	0.554	0.814
A	0.864	0.697	0.771	0.917
SPAN (leaves)	0.911	0.952	0.931	0.940
Weighted Average	0.782	0.782	0.775	0.844

C. Result discussion

We choose some pages as the examples to analyze the generating tags. From table 2 we can see the tag <P> has a very low accuracy before we do the manually correction. This is because most of these elements are grouped to the class DIV, and in this case, this classification is accepted as Fig. 3(a) shows. The low accuracy of <FORM> has the same reason that some parent tag of <INPUT> is <DIV> but we change them to be the <FORM> which is also acceptable.

Some of our result has improved the structure as Fig. 3 (b) shows, in this case, the original tag of the element is <A> but it's a title obviously, thus our method gives it a tag <H>. Same case can be found in Fig. 3(c) that the original tag is but it's a single tag with a line of text, the predicting tag is <A>.

Also our results have some mistakes, as Fig. 3 (d) and (e) shows. In Fig. 3(d), the element should be an image with tag , but our method gives it a tag of <P>. It's because our method count the number of main color by counting the clusters of colors in the image, which returns two colors and the method thought it's a paragraph of texts. Adding the analysis of fonts with the help of OCR may reduce these mistakes. The lacking of the analysis of fonts causes the similar errors in predicting the titles. Our method doesn't take the properties of font into consideration that it wrongly gives the elements a tag <A> instead of <H>.



(a) the predicting tag is <DIV> and the original tag is <P>

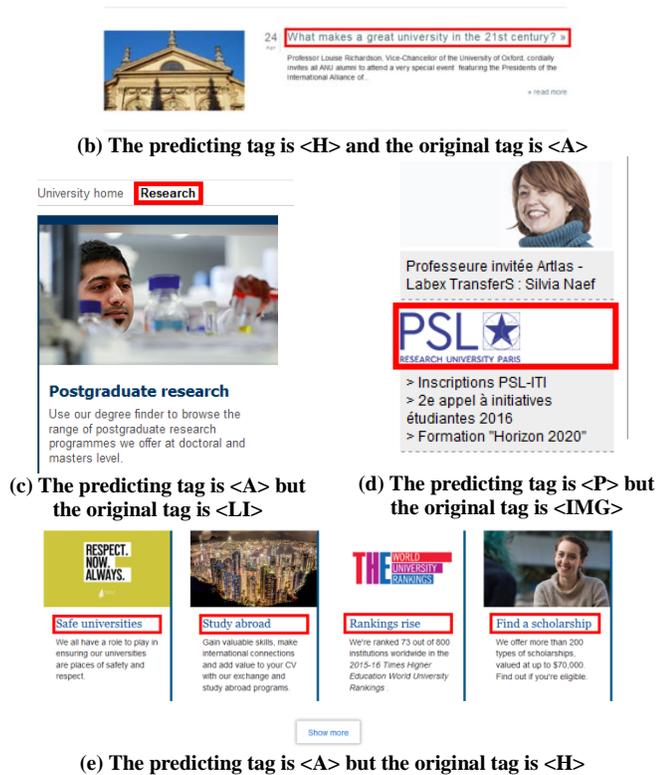


Figure 4. Some inconsistencies between the generated tags and the element's actual tags (a) both of the tags are acceptable; (b),(c), our results are acceptable but the actual tags are wrong; (d),(e) our results are wrong.

The high accuracy of leaves with tag <A> and has two reasons; the first is our method can find a line of text effectively, and the second is most leaves are texts in our selected web pages.

The results show that in most cases, our method can generate convincing tags for elements. But the analysis of fonts could be added to reduce mistakes in the generating of some tags of texts and logos. And the analyzing of color features should be refined.

CONCLUSION AND FUTURE WORK

In this paper, we propose a method which can generate a web page from the mockup. The bottom-up tag generating method is proved to be correct by the experiment. Future work could extend to make the generating layout to be the responsive layout to meet the requirements of numerous kinds of devices. The current generating web page is static and some basic interactive features could be added in it. Furthermore, some strategies will be used to improve the accuracy of the generating tags.

ACKNOWLEDGMENT

This research is supported by the NSFC Guangdong Joint Fund (No. U1201252), the Science and Technology Planning Project of Guangdong Province (No. 2014B010110003), and the Research Project of Educational Commission of Guangdong Province (No. 2013CXZDB001).

REFERENCES

- [1] Kumar Ranjitha, Arvind Satyanarayan, Cesar Torres, Maxine Lim, Salman Ahmad, Scott R. Klemmer, and Jerry O. Talton. "Webzeitgeist: Design mining the web." In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, pp. 3083-3092. ACM, 2013.
- [2] Myers B, Hudson S E, Pausch R. Past, present, and future of user interface software tools[J]. ACM Transactions on Computer-Human Interaction (TOCHI), 2000, 7(1): 3-28.
- [3] Sushil Bajracharya, Trung Ngo, Erik Linstead, Yimeng Dou, Paul Rigor, Pierre Baldi, and Cristina Lopes, "Sourcerer: a search engine for open source code supporting structure-based search." Proceedings ACM SIGPLAN Conference on Object-Oriented Programming, Systems, Languages, and Applications 2006, pp. 682-682.
- [4] Wing-Kwan Chan, Hong Cheng, and David Lo, "Searching connected API subgraph via text phrases." pp. 1-11 in Proceedings of the ACM SIGSOFT 20th International Symposium on the Foundations of Software Engineering, (2012).
- [5] Reiss, Steven P. "Seeking the user interface." Proceedings of the 29th ACM/IEEE international conference on Automated software engineering. ACM, 2014, pp. 103-114.
- [6] Steven P. Reiss, "Semantics-based code search," International Conference on Software Engineering 2009, pp. 243-253 (May 2009).
- [7] Bjorn Hartmann, Leith Abdulla, Manas Mittal, and Scott R. Klemmer, "Authoring sensor based interactions through direct manipulation and pattern matching," Proceedings of CHI 2007: ACM Conference on Human Factors in Computing Systems, pp. 145-154 (2007).
- [8] Lee, B., Srivastava, S., Kumar, R., Brafman, R., Klemmer, S. R. Designing with interactive example galleries. Proc. CHI (2010), ACM.
- [9] Kumar, R., Talton, J. O., Ahmad, S., & Klemmer, S. R. (2011, May). Bricolage: example-based retargeting for web design. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (pp. 2197-2206). ACM.
- [10] Dong, Mian, and Lin Zhong. "Chameleon: a color-adaptive web browser for mobile OLED displays." Mobile Computing, IEEE Transactions on 11.5 (2012): 724-738.
- [11] Li, Ding, Angelica Huyen Tran, and William GJ Halfond. "Making web applications more energy efficient for OLED smartphones." Proceedings of the 36th International Conference on Software Engineering. ACM, 2014.
- [12] Flatla, D. R., Reinecke, K., Gutwin, C., & Gajos, K. Z. (2013, April). SPRWeb: Preserving subjective responses to website colour schemes through automatic recolouring. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (pp. 2069-2078). ACM.
- [13] Chen, X., Long, Y., & Luo, X. (2015). Automatic Color Modification for Web Page Based on Partitional Color Transfer. In Software Reuse for Dynamic Systems in the Cloud and Beyond (pp. 204-220). Springer International Publishing.
- [14] Cai, D., Yu, S., Wen, J. R., & Ma, W. Y. (2003). VIPS: a vision-based page segmentation algorithm (p. 28). Microsoft technical report, MSR-TR-2003-79.
- [15] Rossi G, Urbietta M, Ginzburg J, et al. Refactoring to rich internet applications. A model-driven approach[C]//Web Engineering, 2008. ICWE'08. Eighth International Conference on. IEEE, 2008: 1-12.
- [16] Sánchez Ramón, Ó., Sánchez Cuadrado, J., & García Molina, J. (2010, September). Model-driven reverse engineering of legacy graphical user interfaces. In Proceedings of the IEEE/ACM international conference on Automated software engineering (pp. 147-150). ACM.
- [17] Akiki, Pierre A., Arosha K. Bandara, and Yijun Yu. "Adaptive model-driven user interface development systems." ACM Computing Surveys 47.1 (2015).
- [18] <http://research.defool.me/appprofiler/>

An Empirical Study to Evaluate the Feasibility of a UX and Usability Inspection Technique for Mobile Applications

Ingrid Nascimento, Williamson Silva, Adriana Lopes, Luis Rivero, Bruno Gadelha, Elaine Oliveira, Tayana Conte
Federal University of Amazonas (UFAM)
Manaus, Brazil
{inc, williamson.silva, adriana, luisrivero, bruno, elaine, tayana}@icomp.ufam.edu.br

Abstract — Usability and UX (User eXperience) are some of the most important factors for evaluating the quality of mobile applications. They focus on how easy to use an application is and the emotions that such use evokes. However, these aspects are often evaluated separately in industry through different evaluation techniques. Although it is possible to identify more usability and UX problems by employing different UX and usability evaluation methods, this distributed approach may not be cost effective and may not allow to thoroughly explore the identified issues. In order to support the identification of both UX and usability problems in a single evaluation, we have proposed Userbility, an UX and usability inspection technique that allows evaluating these aspects in mobile applications. This paper presents an empirical study over the second version of Userbility to verify its feasibility. In this study, we compared Userbility with the UX and Usability Guidelines Approach (UUGA) that helps the evaluation of usability and UX separately in mobile applications. According to the quantitative results, considering efficiency, UUGA was better than the Userbility technique. However, the qualitative results suggest that Userbility pointed more improvement suggestions, which could be useful for redesigning the evaluated application.

Keywords- *User experience; usability; mobile applications; evaluation; inspection; empirical study.*

I. INTRODUCTION

In the past few years, the development of mobile applications has grown considerably due to growth of the mobile devices market [1]. Therefore, to help the quality of the developed mobile applications, it is necessary to carry out specific evaluations considering their dynamism and the aspects that make it difficult to evaluate them (e.g mobile context, connectivity, size of the screen and others) [2].

Usability is one of the factors for the adoption of mobile applications [2]. According to the ISO 9241-210 norm [3], usability is the extent to which “*a product can be used by specified users to achieve specified goals with effectiveness, efficiency and satisfaction in a context of use*”. Moreover, another factor related to quality in use is User eXperience (UX). UX considers the quality of an application regarding the feelings of the user when interacting with it. This term denotes the overall experience of the user of an application [4], and UX is crucial for the adoption of an application with success [4]. In this sense, while usability focuses on the ease of carrying out tasks and overall satisfaction, UX focuses on aspects related to

the emotions, perceptions and judgements of an application. Therefore, software development teams willing to increase the quality in use of the developed mobile applications need to evaluate both of them.

To evaluate usability and UX together, in our previous work [13], we developed Userbility in order to support inspectors in the evaluation of both UX and usability in mobile applications at the same time, to assess whether Userbility can support inspectors in detecting usability and UX problems. Nascimento et al. [13] conducted a study with five mobile applications. The results showed that it is possible to identify improvements in applications, and allowed us to identify problems during the use of the technique. Based on this, in this paper, we proposed a new version of the technique and an empirical study to evaluate the feasibility of Userbility. We have compared the Userbility to an approach proposed by De Paula et al. [5], which evaluates UX and usability separately.

The remainder of this paper is organized as follows. Section II presents a background on UX and usability evaluation techniques that can be applied to evaluate mobile applications. Then, Section III shows the Userbility technique in its second version. Section IV presents the empirical study where we compared Userbility with another evaluation approach. In Section V, we present the results of the empirical study. Finally, Section VI presents our conclusions and future work.

II. BACKGROUND

Traditional methods for evaluating usability and UX may not be adequate for evaluating mobile applications since mobile devices offer a new paradigm for Human Computer-Interaction (HCI) that these traditional evaluation methods do not consider [2]. Therefore, specific methods for the evaluation of mobile devices may allow providing more beneficial data for mobile applications than those of traditional evaluation [2]. The following are some methods for such purpose.

Von Wangenheim et al. [6] proposed the Checklist Heuristic Evaluation for Smartphones Applications. This checklist has 12 heuristics for evaluating applications. Each heuristic has usability items that allow measuring the traditional usability of mobile applications.

Another approach is the Expressing Emotions and Experiences (3E), which is a method that aims to capture the experience and feelings of the user [7]. In 3E, the user is

provided with a simple pictorial template for expressing emotions and experiences.

Wetzinger et al. [8] proposed a process for evaluating UX and usability comparing the use in tablets and laptops. This process was composed of the following approaches: the SUS (Software Usability Scale) questionnaire, to measure usability through scales; the AttrakDiff to measure UX; and the SEQ (Single Ease Question), to measure ease of use.

De Paula et al. [5] suggested an approach that uses guidelines to evaluate UX and usability of mobile application by employing Design Thinking, called UX and Usability Guidelines Approach (UUGA). This approach is based on: (a) Usability guidelines by Gong and Tarasewich [9], guidelines by Shneiderman [10], the ISO 9241-210 [3] and Guidelines by Nielsen [11]; and (b) UX guidelines, based on the questionnaire by Chen and Zhu [12], where scores are given to each criterion that influences the experience of the user.

Regarding the methods, Von Wangenheim et al. [6] focuses only in usability evaluation, while 3E method focuses in UX evaluation. Also, the process proposed by Wetzinger et al. [8] did not carry out the inspection of an application; it only uses SUS for measuring usability through a five-point scale, ranging from totally agree to totally disagree for about 10 questions. De Paula et al. [5] suggested the use of an approach to evaluate UX and usability separately in mobile applications. In this approach, the identified UX and usability problems are not directly associated, which could generate repetition of information for the analysis and could not associate the problems to the possible improvements captured with UX evaluation.

III. THE USERBILITY TECHNIQUE

Userbility v 1.0 (Integrating **User** eXperience and **Usability**) was proposed to allow the evaluation of the UX and usability of mobile applications by unexperienced inspectors in the area of HCI [13]. We improved the Userbility v 1.0 based on the methods Checklist Heuristic Evaluation for Smartphones Applications [6] and 3E [7]. We chose these methods as they provide effective means to evaluate specific usability attributes of mobile applications [6] and allow gathering rich emotional responses from users, thus complementing others methods. The integration of these methods aims to make it easier and faster to evaluate UX for practitioners that are not experts in carrying out UX and usability evaluation. Figure 1 shows an example of Userbility in its second version.

Item 1 (Usability Heuristic) shows one of the twelve Usability Heuristics for mobile applications that were included to support the inspection process. We followed the suggestions by Von Wangenheim et al. [6], including usability items for the heuristics so that they could guide the inspectors in finding usability problems. Item 2 (Usability items and an Examples) presents the Usability items and an example for each item. Items 3 (UX Q1) and 4 (UX Q2) present two UX questions with an example for each of them. These questions were included considering the 3E method, in order to support the evaluator when reporting his/her experience for each aspect.

The two UX questions are: (Q1) “How did you react towards the application regarding this aspect? Why did you react that way?”, and (Q2) “What do you think or would improve about this aspect? Point where it should be improved in the application”. Item 5 (Satisfaction Item) shows an item related to the degree of satisfaction of the evaluator. This item was adapted from the Self-Assessment Manikin [14] and is composed of a 5-point scale (very unsatisfied, unsatisfied, neither satisfied nor unsatisfied, satisfied and very satisfied), where a human face depicting an emotion represents each item. With this item, we intend to obtain more information about how much the evaluator was satisfied regarding the evaluated usability heuristic. Figure 1 also shows an example of application and a part of inspection, in Problems (P1 and P2) and Improvements box.

Figure 1. Example using one of the heuristics of Userbility.

IV. THE EMPIRICAL STUDY

We conducted the empirical study in order to compare Userbility v 2.0 with the UX and Usability Guidelines Approach (UUGA) [5], to verify to what extent it could improve the performance of the evaluators in terms of identified problems, time and improvements suggestions to the problems. We chose UUGA due to the following reasons: the evaluator does not need any experience to use the method and it also allows identifying and describing usability problems.

A. Planning

During the planning stage, we defined the hypotheses of the experiment, its context, the selection of subjects and their training in the techniques, and the tasks to be carried out.

1) **Hypotheses:** the experiment was conducted to test the following hypotheses: H_{01} – There is no difference in terms of efficiency regarding Userbility and UUGA; H_{A1} – There is a difference between the efficiency regarding Userbility and UUGA; H_{02} – There is no significant difference in terms of effectiveness regarding Userbility and UUGA; H_{A2} – There is a significant difference between the effectiveness regarding Userbility and UUGA. In this sense, effectiveness is defined as “the percentage of identified defects divided by total (known) defects”, and efficiency is defined as “the total number of defects found by an inspector divided by the total time spent to find them” [15].

2) **Context and Subjects:** We conducted the study at Federal University of Amazonas (Brazil) using both approaches (Userbility and UUGA) to evaluate a mobile application called “In the Tip of the Tongue” (Na Ponta da Língua – in Portuguese). “In the Tip of the Tongue” is an application that aims to help students when they are learning Portuguese, by describing the origin of the words and helping users fix their meaning through a simple game. At all, 49 students from two courses on Human Computer-Interaction (HCI) and Collaborative Systems agreed to participate in the study as evaluators. All subjects signed a consent form to participate in the study and filled out a characterization questionnaire to measure their knowledge in HCI (KH) and their knowledge in software Analysis and Design (KA). The characterization form was employed to categorize the subjects as having: (a) No Experience – does not have knowledge in HCI or possesses some notions on usability acquired through lectures/ readings, but without practical experience; (b) Low – participated in at least one usability evaluation/ project in class; (c) Medium – participated in 1 to 4 usability evaluation/ projects in industry; (d) High – participated in 5 usability project/ evaluations in industry. The expertise on KA was classified following the same standards. From this characterization, we divided the subjects into two groups to carry out the inspection: the group from the Userbility was formed of 25 subjects and the UUGA by 24 subjects. The group of UUGA had one more subject with high knowledge degree in software analysis and design when compared to the Userbility group (see Table I – third and fourth columns).

3) **Training:** We trained the subjects introducing the concepts and examples of Usability and UX. In addition, they had exercises in which they had to give their impressions on the usability and UX of one application of their choice. After balancing the subjects, we randomly distributed them into two groups where each group received equivalent training on each of the techniques that they would apply.

4) **Tasks definition:** Each of the evaluators had to carry out six main tasks during his/her evaluation of the application. The tasks were: (a) To train, answering at least three training questions; (b) To start a game, choose the correct word; (c) To view the final score in the game; (d) To search for a word in the dictionary; (e) To view the graph of the languages; and (f) To find information about questions of Portuguese language.

We selected the tasks based on their importance within the application to achieve user goals.

B. Execution

Following their characterization, the subjects then received a tablet or cell phone to evaluate the selected application using one of the two techniques. When starting the evaluation, each subject received an introduction to the evaluated application and the list of tasks they would employ to evaluate its usability and UX individually. Each group carried out the evaluation in separate rooms with a researcher who supervised all subjects. After the evaluation, we grouped the discrepancies (possible usability problems indicated by an inspector) that were reported by each of the techniques in a single list. In this list, we removed the duplicated discrepancies from the same inspector before the discrimination of the discrepancies. After that, the list of discrepancies was reviewed by another researcher related to this study to avoid misclassification.

C. Analysis

We carried out a discrimination meeting to classify the discrepancies into real problems and false positives (discrepancies which were not problems affecting the usability and the UX). The client (owner of the application), a UX and usability expert and three other researchers participated in this meeting, which lasted for 1 hour and 34 minutes.

V. RESULTS OF THE EMPIRICAL STUDY

We obtained the quantitative data from measuring the time, number of real problems and false positives for each subject, and the qualitative data from analyzing the description of the problems found in the Userbility technique inspection questionnaire.

A. Quantitative Analysis

The results from the discrimination meeting have been used to calculate the effectiveness and efficiency of the subjects. Table I shows the results per subject and per technique in terms of discrepancies, identified real problems and time spent to find them. In Table I, we can see that subjects who used Userbility managed to find between 1 and 11 defects spending about 0.47 and 2 hours. On the other hand, the subjects that used UUGA employed between 0.22 and 1.05 hours, however they also found between 1 and 11 defects. The subjects took more time using Userbility according to the applied statistical tests. Table II shows the results per technique in terms of identified problems, unique defects and suggested improvements. The analysis of these results shows that UUGA allowed finding more real defects (NRD). However, among the identified real problems, Userbility allowed identifying more unique defects (total of 37 unique defects - NUD) than UUGA (NUD is 31). Also, we noticed that Userbility allowed pointing out less false positives. This is an important feature, as accurate techniques (those in which the chances of a discrepancy become a real problem is high) are important for identifying more problems.

To test the hypotheses defined in Subsection IV-A, we carried out a statistical analysis using the SPSS tool ($\alpha = 0.05$). We used the Shapiro-Wilk test to check the normality. This test is indicated for sample with size less than 50 [16]. We verified

that efficiency ($p = 0.06$ for the Userbility and $p = 0.08$ for the UUGA) and effectiveness ($p = 0.72$ for the Userbility and $p = 0.36$ for the UUGA) were normally distributed. Based on the results, we applied the *t-test* [17]. Figure 2 shows the boxplots graphs to facilitate the visualization of the results.

TABLE I. SUMMARY OF INSPECTION RESULT PER SUBJECT.

T	#S	KH	KA	ND	NFP	NRD	Ti(h)	Effic.	Effec.(%)
Userbility	S01	N	N	3	0	3	1.42	2.12	4.41
	S02	N	N	2	0	2	0.83	2.40	2.94
	S03	N	N	5	0	5	2.00	2.50	7.35
	S04	L	N	5	1	4	0.60	6.67	5.88
	S05	L	N	14	3	11	1.80	6.11	16.18
	S06	L	N	4	2	2	0.83	2.40	2.94
	S07	L	L	10	1	9	1.50	6.00	13.24
	S08	L	L	3	2	1	0.75	1.33	1.47
	S09	L	L	4	0	4	0.92	4.36	5.88
	S10	L	L	5	1	4	0.67	6.00	5.88
	S11	L	L	9	4	5	1.28	3.90	7.35
	S12	L	L	7	1	6	0.92	6.55	8.82
	S13	L	L	7	0	7	0.80	8.75	10.29
	S14	L	L	6	3	3	0.83	3.60	4.41
	S15	L	L	6	1	5	1.15	4.35	7.35
	S16	M	L	3	0	3	0.78	3.83	4.41
	S17	M	L	2	0	2	0.50	4.00	2.94
	S18	M	M	4	1	3	-	-	4.41
	S19	M	M	4	0	4	1.00	4.00	5.88
	S20	L	L	6	1	5	0.53	9.38	7.35
	S21	L	N	4	0	4	1.65	2.42	5.88
	S22	L	L	4	0	4	0.50	8.00	5.88
	S23	N	L	7	0	7	1.00	7.00	10.29
	S24	L	M	6	1	5	-	-	7.35
	S25	H	H	8	0	8	0.47	17.14	11.76
S26	N	N	3	0	3	0.55	5.45	4.41	
S27	N	N	7	3	4	0.50	8.00	5.88	
S28	L	N	3	0	3	0.77	3.91	4.41	
S29	L	N	5	1	4	0.52	7.74	5.88	
S30	L	N	5	2	3	0.67	4.50	4.41	
S31	L	N	4	1	3	0.65	4.62	4.41	
S32	L	L	8	3	5	0.78	6.38	7.35	
S33	L	L	11	2	9	1.05	8.57	13.24	
S34	L	L	8	2	6	0.60	10.00	8.82	
S35	L	L	5	1	4	0.53	7.50	5.88	
S36	L	L	7	1	6	0.43	13.85	8.82	
S37	L	L	4	1	3	0.75	4.00	4.41	
S38	L	L	9	1	8	0.97	8.28	11.76	
S39	L	L	4	0	4	0.52	7.74	5.88	
S40	L	L	3	1	2	0.77	2.61	2.94	
S41	M	L	3	2	1	0.58	1.71	1.47	
S42	L	M	7	1	6	0.45	13.33	8.82	
S43	M	M	10	0	10	0.90	11.11	14.71	
S44	M	H	5	2	3	0.22	13.85	4.41	
S45	L	L	3	1	2	0.43	4.62	2.94	
S46	N	N	2	0	2	0.85	2.35	2.94	
S47	N	L	5	0	5	0.23	21.43	7.35	
S48	L	L	12	1	11	0.47	23.57	16.18	
S49	H	H	10	3	7	0.77	9.13	10.29	

Footnote - T: Technique; **#S:** Subject; **KH:** Knowledge in HCI; **KA:** Knowledge in software Analysis and Design; **H:** High; **M:** Medium; **L:** Low; **N:** None; **ND:** Number of Discrepancies; **NFP:** Number of False Positives; **NRD:** Number of Defects; **Ti(h):** Time; **Effic.:** Defects per Hour; **Effec.(%):** Effectiveness.

Figure 2 (Efficiency) presents the boxplot graphs comparing the efficiency indicator. When analyzing the median of both groups, we can see that the group applying UUGA had higher efficiency than the group that applied the Userbility. The p -value (0.009) $<$ 0.05 rejects H_{01} and supports H_{A1} , which indicates that evaluators using UUGA are more efficient than those applying Userbility. A possible cause for this result may

be the number of items that evaluators had to fill in, when applying Userbility.

TABLE II. ANALYSIS OF THE DISCREPANCIES PER TECHNIQUE.

Technique	ND	NFP	NRD	NUD	SI
Userbility	138	22	116	37	190
UUGA	143	29	114	31	11
Total	246	45	201	68	201

Footnote - UUGA: UX and Usability Guidelines Approach; **ND:** Number of Discrepancies; **NFP:** Number of False Positives; **NRD:** Number of Defects; **NUD:** Number of Unique Defects; **SI:** Improvements Suggested.

Figure 2 (Effectiveness) presents the boxplot graph comparing the effectiveness indicator. When analyzing the medians, we can see that the effectiveness of the Userbility group was almost the same as the one of the group that applied UUGA. The p -value (0.886) $<$ 0.05 supports H_{02} , which indicates that is not possible to make statements about effectiveness of Userbility and UUGA. It is important to note that Userbility allows finding more unique (not repeated) defects and can support identifying improvements based on the perceptions of the subjects, which is an important feedback for the correction of problems and redesigning the evaluated application. The subjects S25 and S48, that had high and low skill level, were more efficient for the approaches. And the subjects S05 and S48, that had low and none skill level, were more effective for the approaches. This indicates that the evaluator's skills levels may not influence in the effectiveness.

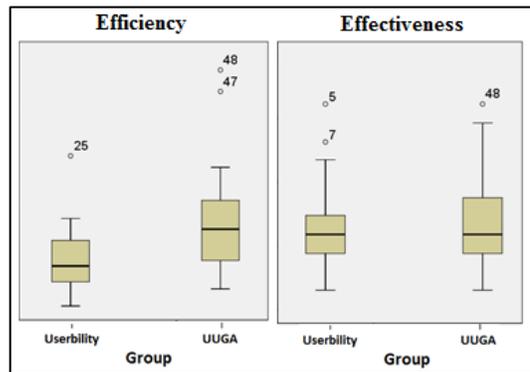


Figure 2. Boxplot graphs for the efficiency and effectiveness indicators.

B. Qualitative Analysis

The analysis of the qualitative data began with the examination of the answers within the Userbility technique questionnaire. The number of improvements suggested to the Userbility (190) was much higher than the number of improvements suggested to the UUGA (11), showed in the Table II. The improvements suggested by the subjects were related to the problems identified in the application. The improvements suggestions to some problems were:

The application stopped without warning – “The app should have a message about what happened...” (S04);

Difficulties in understanding the symbols – “They could add a help button about the features of the app” (S07);

Many steps to perform the task to find information about questions (task f in the Section IV) – “The app could have a review option about the performance of some tasks...” (S07);

The graph didn't provide any relevant information for the application – *When we touch the screen and we choose Greek language, for instance, there could be some fun facts about such language.*" (S13);

More feedback to the user in the game – *"There could be some way to alert that the bubbles would appear and how to play, since I failed to notice that I should choose the correct word."* (S18).

These improvements suggestions were useful for redesigning the application and improving the quality of the application. And they also could be useful to improve the UX, in order to generate the successful adoption of the application.

VI. CONCLUDING REMARKS AND FUTURE WORK

This paper described and evaluated the second version of Userbility, a technique to integrate and facilitate UX and usability inspections in mobile applications. In our empirical study, we compared the Userbility with the UX and Usability Guidelines Approach (UUGA). Among the results, the statistical analysis suggested that Userbility was less efficient than UUGA. Nevertheless, when analyzing the results from Table II, we noticed that Userbility allowed inspectors to point out less false positives. This is an important feature as the degree of accuracy of a technique can avoid inspectors to waste their time in indicating problems not affecting the quality of the evaluated application. In addition, Userbility allowed inspectors to point out more unique problems and improvements suggestions, which can be useful for redesigning the evaluated application. Furthermore, this empirical evaluation showed evidence of its feasibility.

We considered also the threats to validity in this empirical study. The main threats to validity were: (1) training effects: we controlled this risk by providing equivalent training for both approaches; (2) students are probably not good substitutes for professional inspectors: since we were looking for novice inspectors with no knowledge on the use of the applied techniques; (3) academic environments do not represent day to day experience in the industry: we carried out the evaluation over a real mobile application which can help resemble a real industry environment; (4) homogeneity of the sample: there is a limitation regarding our study, even though the study was carried out in classes from different courses; (5) the subjects did not use the same device: not using the same device for inspecting the application could cause bias to our results, but we needed to identify problems in both devices to verify the correctness of the application in different platforms.

As future work, we intend to carry out improvements in the Userbility technique to enhance its effectiveness and efficiency when compared to other inspection approaches. Also, we intend to replicate this study to draw further conclusions on the feasibility and applicability of Userbility in industrial environments.

ACKNOWLEDGMENT

We thank the financial support granted by CAPES process 175956/2013 and FAPEAM through processes numbers: 062.00600/2014; 062.00578/2014. Furthermore, we would like to thank all the volunteers who participated in the study. We also thank Natasha Valentim for her remarks on this research.

REFERENCES

- [1] IDC, Smartphone OS Market Share, Q1 2015. Available at: <http://www.idc.com/prodserv/smartphone-os-market-share.jsp>, 2015.
- [2] D. Zhang and B. Adipat, "Challenges, methodologies, and issues in the usability testing of mobile applications," in *International Journal of Human-Computer Interaction*, vol. 18, n. 3, pp. 293-308, 2005.
- [3] International Organization for Standardization, ISO FDIS 9241-210: Ergonomics of human system interaction – Part 210: Human-centered design for interactive systems (formerly known as 13407). Switzerland, 2009.
- [4] A. Charland and B. Leroux, "Mobile application development: web vs. native," in *Communications of the ACM*, vol.54, n.5, pp. 49-53, 2011.
- [5] D. F. De Paula, B. H. Menezes and C. C. Araújo, "Building a Quality Mobile Application: A User-Centered Study Focusing on Design Thinking, User Experience and Usability," in *Part of HCI International 2014. Design, User Experience, and Usability. User Experience Design for Diverse Interaction Platforms and Environments*, vol. 8518, pp. 313-322, 2014.
- [6] C. G. Von Wangenheim, T. A. Witt, A. F. Borgatto, J. V. Nunes, T. C. Lacerda, C. Krone and L. de Oliveira Souza, "A Usability Score for Mobile Phone Applications Based on Heuristics," *International Journal of Mobile Human Computer Interaction*, vol. 8, n. 1, pp. 23-58, 2016.
- [7] M. Tähti and M. Niemelä, "3e-expressing emotions and experiences," in *Workshop on Innovative Approaches for Evaluating Affective Systems. HUMAINE*, 2006, pp. 15-19.
- [8] W. Wetzlinger, A. Auinger and M. Dörflinger, "Comparing effectiveness, efficiency, ease of use, usability and user experience when using tablets and laptops" in *Part of HCI International 2014. Design, User Experience, and Usability. Theories, Methods, and Tools for Designing the User Experience*, vol. 8517, pp. 402-412, 2014.
- [9] J. Gong and P. Tarasewich, "Guidelines for handheld mobile device interface design," in *Annual Meeting, Northeastern University, Boston*, 2004, pp. 3751-3756.
- [10] B. Shneiderman, "Designing the User Interface - Strategies for Effective Human-Computer Interaction," Addison-Wesley, 1998.
- [11] J. Nielsen, "Usability inspection methods," in *Conference Companion on Human factors in computing systems*, 1994, pp. 413-414.
- [12] Z. Chen and S. Zhu, "The Research of Mobile Application User Experience and Assessment Model," in *International Conference on Computer Science and Network Technology*, vol. 4, pp. 2832-2835, 2011.
- [13] I. Nascimento, W. Silva, B. Gadelha, T. Conte, "Userbility: A Technique for the Evaluation of User Experience and Usability on Mobile Applications," in *18th International Conference on Human-Computer Interaction*, in press.
- [14] M. M. Bradley and P. J. Lang, "Measuring emotion: the self-assessment manikin and the semantic differential," *Journal of behavior therapy and experimental psychiatry*, vol. 25, n. 1, pp. 49-59, 1994.
- [15] A. Fernandez, S. Abrahão and E. Insfran, "Towards the validation of a usability evaluation method for model-driven web development," in *International Symposium on Empirical Software Engineering and Measurement*, 2010, pp. 54-57.
- [16] S. Shapiro and M. Wilk, "An Analysis of Variance Test for Normality (Complete Samples)," *Biometrika*, vol. 52, pp. 591-611, 1965.
- [17] N. Juristo and A. M. Moreno, *Basics of Software Engineering Experimentation*. Boston, MA: Kluwer Academic Publishers, 2001.

How Novice Software Engineers Apply User Interface Design Patterns: An Empirical Study

Luis Rivero and Tayana Conte

USES Research Group, Instituto de Computação, Universidade Federal do Amazonas
Manaus – Brazil
{luisrivero, tayana}@icompu.ufam.edu.br

Abstract— Conveying a positive User eXperience (UX) is essential for the success of any application, as it affects users' intentions to use a product. Nevertheless, novice software engineers may have difficulties in finding an appropriate solution for UX problems due to their lack of knowledge in interface design. User interface design patterns are well-working solutions to user problems that occur repeatedly which can help new developers solve interface problems. Although several studies investigated the effectiveness of applying design patterns in real development projects, few studies evaluate the difficulties that novice software engineers face when applying user interface design patterns for the first time. To understand how novice software engineers apply design patterns in the correction of UX problems, we carried out an empirical study with four small novice development teams redesigning mobile applications interfaces. We collected data on factors affecting the applicability of the employed design patterns, through questionnaires. Our results suggest that novice software engineers find this type of patterns useful but require means to find and select an adequate pattern to solve a problem.

Keywords— design patterns; interface design; user experience; usability; software quality; mobile application, empirical study

I. INTRODUCTION

User eXperience (UX) has emerged as a new way of understanding and studying the quality in use of interactive products [1]. It is a consequence of a user's internal state, the characteristics of the designed system and the context within which the interaction occurs [2]. When developing a mobile application, UX plays a central role as applications conveying a positive UX are more likely to be adopted rapidly and gain acceptance [3]. Consequently, to convey a positive experience, it is necessary that the mobile application provides functionality and easy means to access it [4].

User interface design patterns can be a cost-effective alternative for correcting UX problems, as they describe best-practice solutions to problems faced by users during their interaction with an application [5]. These patterns also contain a discussion of when it is appropriate to apply the proposed solutions and what the consequences may be [6]. However, even though several authors evaluated the use of design patterns [6][7][8], few evaluate how novice software engineers deal with them while designing an interface. A proper understanding of the applicability of these patterns by software engineers is necessary to evaluate their feasibility in industry.

Qualitative methods are an alternative to better comprehend the issues that need a more specific and detailed analysis. They

are recommended when it is necessary to consider human behavior in a research or when we need to understand the factors that affect the object of study [9]. In this paper, we carried out an empirical study using qualitative methods to better understand how novice software engineers apply design patterns and what makes it easy or difficult to employ them during the redesign of a mobile application. To this end, four teams of novice software engineers suggested modifications in the interface of two real mobile applications, using interface design patterns. Here, we present our findings regarding such use in terms of the factors that affected the applicability of the user interface design patterns.

The remainder of this paper is organized as follows. Section II presents a brief review on user interface design patterns and studies regarding their feasibility. Then, Section III presents the steps for carrying out the empirical study, while Section IV presents the analysis of its results. Finally, Section V presents a comparison of our findings with the ones found in literature and Section VI presents our conclusions and future work.

II. DESIGN PATTERNS FOR USER INTERFACE DESIGN

User interface design patterns have been introduced as a medium to capture and represent solutions to users' problems when interacting with a software application [8]. When designing software applications that provide a positive UX, user interface design patterns are tools that can provide an alternative to abstract and reuse the essential details of successful and usable interfaces, providing a positive UX.

A. Patterns Proposals for User Interface Design

A number of generic pattern languages have been proposed for suggesting both specific interface layouts and interaction steps. For instance, Tidwell [10] proposed a Pattern Language intended to support high-quality interaction between a person and a software artifact, aimed at passive activities (e.g. absorbing information with little or no interactivity) to hands-on activities (e.g. creation of other objects). Additionally, Welie [11] proposed a pattern library which contains best practices with examples and insights on their applicability.

Patterns that are proposed for the design of specific types of applications (e.g. web, mobile, games, others) can be more effective in guiding software engineers in the correction of UX problems [12]. Consequently, specific patterns have emerged as a means to facilitate the development of mobile applications meeting users' needs. For instance, some patterns have been proposed by software companies (e.g. Apple, Google,

Microsoft) to support developers in designing applications for specific operating systems, while other patterns have been proposed for mobile devices in general. Examples of each of these types of patterns can be found elsewhere [13][14].

B. Evaluation of User Interface Design Patterns

According to Koukouletsos et al. [8], the potential of patterns and patterns languages has been widely promoted, but there has been little empirical evidence about pattern use and its benefits for interface design. Among the studies evaluating the use of design patterns in such context, Chung et al. [6] evaluated the effectiveness of their proposed design patterns for the design of ubiquitous applications. Furthermore, Lanzilotti et al. [7] studied how a pattern-based inspection approach could assist novice software engineers in the evaluation of interfaces. Additionally, Koukouletsos et al. [8] verified the difference between applying design patterns and design guidelines.

Even though the above studies evaluated the use of design patterns, few have focused on both patterns for mobile applications interface design and their applicability by novice software engineers. For instance, in the study by Chung et al. [6] some of the participants were experienced software engineers with knowledge in UI and Web design. Also, Lanzilotti et al. [7] did not focus on the effect of applying design patterns for (re)designing an application. Additionally, the evaluation by Koukouletsos et al. [8] was not performed on mobile applications. Consequently, there is still need to gather information on how novice software engineers apply mobile patterns in the (re)design of mobile applications interfaces. Such information will be useful for identifying improvement opportunities in the application of design patterns by novice software engineers in the industry, and for improving the UX of the developed applications.

III. THE EMPIRICAL STUDY

We conducted the study at a university in the city of Manaus (Brazil) during a class on software quality at a Computer Science course. Some of the topics from the class were: usability/UX and ways of identifying and correcting usability/UX problems. The study took place during 6 weeks out of the 4 month duration class.

A. Participants

At all, 12 software engineering students in their last semester of college were enrolled in the class and agreed to participate in the study. These software engineers had technical background (more than 3 years of experience studying and/or practicing in the area) on software development methodologies, but low or no experience in interface design.

B. Materials and Procedure

We employed a qualitative research methodology aiming to understand which factors contribute or make it difficult to apply design patterns in the redesign of a software interface by novice software engineers. The students were divided into four groups of three novice software engineers each, and each group participated in all of the following stages.

Stage 1 - Lectures and training: At this stage, the novice software engineers participated on regular classes on the methods that were to be applied. A specific training regarding

usability and UX evaluations was prepared as part of the class. Also, this training contained examples of how design patterns had been applied to correct usability and UX problems.

Stage 2 - Usability evaluation on mobile applications in the market: At this stage, each group of novice software engineers chose between two mobile applications for android devices and carried out a first usability inspection using the Heuristic Evaluation, a method that suggests ten rules for identifying usability issues [15]. The evaluated applications were: (a) Manaus Bus¹, an application that supports users in finding the right bus to their destination; and (b) Manaus in Theaters², which helps users find out about the different movies on display in the theaters of the city. We focused on these applications as they are entertainment and information mobile apps³, which require a high degree of usability to facilitate their use and enhance the UX of the users. Therefore, they could be a good example of how redesigning an application could improve its perceived quality in terms of UX [3]. After collecting all inspection reports, the identified problems were verified by a high experienced analyst (with more than 5 years of experience in usability and UX evaluations) to check which problems were real and which were false positives. There was a meeting with each team discussing the inspection results.

Stage 3 - Correction of the identified problems applying user interface design patterns: At this stage, the novice software engineers employed the design patterns for mobile applications [13][14] and proposed changes in the user interface. We made sure that the software engineers knew that they could apply the design patterns as they felt appropriate. This was done in order to understand what process they would use to redesign the application when applying the design patterns and the difficulties they faced during such application.

To collect the qualitative data, right after finishing stage 3, we applied a questionnaire containing open questions to: (a) assess the acceptance of the use of design patterns for the improvement of user interfaces by novice software engineers; and (b) identify advantages, constraints and improvement opportunities for improving the adoption of design patterns in the software industry by novice software engineers. Table I shows each of the asked questions and their purpose. For instance, Q2 asks the novice software engineer which patterns (s)he thought were the easiest and hardest to apply, which aims at making him/her think of the reasons that make a pattern easy or difficult to use, preparing him/her for the next questions.

C. Data Analysis

We analyzed the data obtained from the answers to the questionnaires using procedures from the Grounded Theory (GT) method. GT uses a set of systematic data collection and analysis procedures to generate, prepare, and validate theories on social phenomena or processes [16]. Although the purpose of GT is the construction of theories, a researcher may use only some of its procedures to analyze qualitative data.

According to Strauss and Corbin [16], the coding process, in which concepts (or codes) and categories are identified, can

¹<https://play.google.com/store/apps/details?id=com.manausemcartaz>

²<https://play.google.com/store/apps/details?id=com.onibus.manaus>

³<https://support.google.com/googleplay/android-developer/answer/113475>

be divided into three stages: open, axial, and selective coding. During the open coding, the researcher carries out the breakdown, analysis, comparison, conceptualization, and categorization of the data. During the axial coding, we examine the relations between the categories. The relations between codes can be defined by the researcher him/herself. Finally, the selective coding realizes all the process refinements by identifying the core category with which all others are related.

In our analysis, we have carried out the open and axial coding analyzing the answers to the questionnaires, but have not elected a core category yet because GT suggests the circularity between the collection and analysis stages until the theoretical saturation is reached [16]. Therefore, we do not claim that we applied GT, but some of its procedures as we decided to postpone the selective coding phase.

TABLE I. QUESTIONS FROM THE FOLLOW-UP QUESTIONNAIRE AND THEIR PURPOSE WITHIN OUR STUDY

ID	Open Question	Purpose
Q1	Which were the steps that you followed in order to apply the design patterns?	To identify how novice software engineers apply the design patterns.
Q2	Which was the most difficult design pattern to apply? And which was the easiest to apply?	To make the novice software engineer think on the aspects that make it easy or difficult to apply a design pattern.
Q3	Please, describe the difficulties that you face when searching/understanding/applying the design patterns.	To make the novice software engineers list concrete aspects that made it difficult to apply the design patterns.
Q4	Please, describe the aspects that made it easy to search/understand/apply the design patterns.	To make the novice software engineers list concrete aspects that made it easy to apply the design patterns.
Q5	If you had to redesign an application again, would you consider using design patterns? Why or Why not?	To understand the overall opinion of novice software engineers regarding the use of design patterns for the correction of UX problems.
Q6	What would you do to facilitate the redesign using the design patterns?	To identify improvement opportunities in the use of design patterns for improving the UX of a software application.

IV. RESULTS

We only received the questionnaires from 10 participants even though all 12 were involved in all of the stages. Below, we present our results, referring to the novice software engineers as subjects using the code SXX, where XX is the number of the software engineers from 1 to 10. We have categorized our findings according to the categories that were created from the qualitative data. Table II shows the quotes that from which our finding emerged for each category.

The **Process** and **Steps** categories explain how the novice software engineers applied the design patterns. In our analysis, a process is a collection of steps, but some steps were highlighted as they were incorporated by the novice software engineers themselves. Some subjects described how they applied the design patterns and how they included further stages to facilitate their applications. In this sense, some subjects thoroughly described the exact process (see quote Q01) while others were more direct (see quote Q02). Despite the varying levels of detail in the description of the processes, we identified some specific steps that were incorporated by the

subjects but were not suggested during the training. For instance, some subjects indicated that they searched for other information sources; not only other design pattern proposals but examples of how the design patterns had been applied in real applications to verify their suitability (see quotes Q03 and Q04). Also, some subjects indicated how they selected which problems would be corrected, as well as which pattern they would apply to correct it. In this sense, some of the criteria for prioritizing a problem was based on the difficulty on correcting a problem (see quote Q05) or based on the main functionalities of the app (see quote Q06). Regarding the selection of a pattern, the criteria were: patterns that solve the problem (see quote Q07), patterns that do not make the app difficult to use (see quote Q08) and patterns that are more related to the app's logic (see quote Q09).

Regarding the **Difficulties** and **Facilitators**, the subjects listed several factors that affected the use of the applied patterns. For instance, subjects indicated that applying the user interface design patterns was difficult when (see examples on quotes Q10 and Q11):

- More than one pattern was available and one had to decide which was better
- It was not possible to find a pattern that fit the app
- It was difficult to adapt the solution
- The software engineer had no experience
- The software engineer did not know how to apply the pattern
- The software engineer did not understand the pattern,
- There were few examples of how the pattern was applied

Additionally, the subjects indicated that applying the design patterns was easy when (see examples on quotes Q12 and Q13):

- The explanation of the pattern was simple
- There is training
- There are more examples
- The software engineers knows what (s)he wants to achieve
- There are images, videos and others explaining how to apply them
- The name of the pattern is the same as the problem that needs to be addressed
- One can ask an expert about them
- There are tools which allow support the application process (e.g. tools for searching patterns)
- There are heuristics that help/guide when choosing a pattern.

All subjects indicated that they would apply the design patterns if given the chance (see quote Q14). Among the **Reasons for Employing Design Patterns**, the subjects indicated that they thought the design patterns would guarantee that the application would not be redesigned without following standards (see quote Q15) and that the app would be more visually appealing and organized (see quote Q16). Other aspects that motivated the subjects to employ the design patterns were: providing confidence in the results of the redesign; that the patterns were structured, which made them

easy to apply; that they provide guidance and support; and that they provide solutions to common problems.

Finally, regarding **Improvement Opportunities**, the subjects indicated that patterns could be more specific so they do not become too ambiguous (see quote Q17). Also, they suggested that they themselves study the patterns and the steps for applying them (see quote Q18). This indicates that perhaps, they thought that the training was not enough (or could be improved) for applying the design patterns. Nevertheless, we highlight that we intended to let the subjects apply the design patterns freely, so we could verify what difficulties they encountered and what they included (or would include) in their application process to make it more effective. An interesting finding was regarding the use of the documents we prepared for delivering the activities. In order to collect the data on the redesign process and its results, in Stage 3 (see Section III), we

provided the subjects with a template of a report, which asked the reason for choosing a specific pattern. That report made some of the subjects become more selective of the design pattern that they would apply, as they had to describe the reasons for such choice (see quote Q19). Also, having identified the problems that they had to correct based on an inspection method facilitated identifying the patterns that could correct them, as these selected patterns should made the application meet the heuristic's principles (see quote Q20). This would suggest that having a document guiding the redesign process and a set of rules that need to be followed when deciding which design pattern to use could be introduced into the application process, facilitating the use of the design patterns by novice software engineers. New studies would be necessary to test these hypotheses and evaluate the extent of support that these documents and guidelines could provide for novice software engineers.

TABLE II. QUOTES FROM THE NOVICE SOFTWARE ENGINEERS SUPPORTING OUR FINDINGS WITHIN THE STUDY

ID	Category	Sub	Description
Q01	Process	S06	<i>We had to identify the problems in the application and we searched for other sources containing patterns and possible solutions to the identified problems. We applied the patterns that better solved the problem, following its guidance and we made changes in the interface.</i>
Q02	Process	S08	<i>According to the application we were going to redesign, we chose, as a team, the best option that would fit that application</i>
Q03	Steps	S03	<i>I also looked up additional content, besides the one provided in the android patterns.</i>
Q04	Steps	S09	<i>I verified examples in other applications.</i>
Q05	Steps	S10	<i>We sought what would be easier to correct in the app</i>
Q06	Criteria	S10	<i>We worked on the main functionalities of the manaus bus application</i>
Q07	Criteria	S06	<i>I applied the pattern that better solved the problem" - criteria</i>
Q08	Criteria	S05	<i>I also verified which pattern would be simple enough and would not bother the user or make it difficult to use the app</i>
Q09	Criteria	S04	<i>Understanding the logic of the app (how it navigates, shows content, others) seems to be the best way to choose a pattern</i>
Q10	Difficulties	S03	<i>There was some doubt when applying the navigation pattern, where we would associate the menu option with the help icon, since both were associated with the action bar</i>
Q11	Difficulties	S10	<i>We based on the patterns for mobile applications and we did not find a pattern for that problem</i>
Q12	Facilitators	S04	<i>Yes, because the source where we found the patterns was very simple and direct</i>
Q13	Facilitators	S02	<i>I believe the easiest pattern was the 'links' since there was a pattern with a name that was specific for that</i>
Q14	Intention to use	S07	<i>Yes, i would definitely use them, as they follow the company's standards</i>
Q15	Reason for employing	S05	<i>It helps the software engineer a lot because there is a rule that makes sure that the application is not redesigned without guidance</i>
Q16	Reason for employing	S07	<i>Yes, with the redesign we noticed that it helped a lot in the new layout and made the application much nicer</i>
Q17	Improvement	S01	<i>Some patterns were too repetitive, so they could be made less similar and unambiguous.</i>
Q18	Improvement	S02	<i>I would study more so i could understand them and apply them.</i>
Q19	Influencing Factors	S05	<i>The template of the document helped when carrying out the redesign... i wondered if choosing the specific pattern would correct the problem.</i>
Q20	Influencing Factors	S06	<i>The heuristics [from the heuristic evaluation] were important for identifying the problems, thus they were also the reference for choosing patterns that would not violate them.</i>

ID – Code for the quotation, Category – Category from applying GT procedures, Sub – Subject ID, Description – Literal quotation

V. COMPARISON WITH PREVIOUS RESEARCH

We compared our results to the ones found by other authors to verify similarities or discrepancies. Regarding the work by Lanzilotti et al. [7], we found out that some of the difficulties in applying patterns were corroborated such as the difficulty in matching the correct pattern to a problem. The authors, however, identified that this could make software engineers disregard the correction of some of the identified problems. In our case, some of the software novice engineers indicated that when they did not know which problem they had to correct first, they based their decision on the importance of correcting the problem to achieving user goals.

While Koukouletsos et al. [8] did not provide qualitative results, their quantitative analysis suggests that applying patterns can improve the performance of novice software

engineers. In this sense, the answers from the questionnaires within our study suggest the same, and provide reasons for such improvement such as making the redesign process easier.

Finally, Chung et al. [6] managed to obtain feedback from a subset of the participants who were novice software engineers. In this sense, they identified that participants using design patterns experienced less difficulties and extensively used the patterns in generating ideas and finding solutions. They also found out that another advantage of using the patterns were their usefulness to convey ideas to others within the group. Although our subjects did not report this feature, they indicated that they searched for further patterns in order to find further ideas. Additionally, we need to investigate whether the use of a template for reporting the use of the design patterns may have positively influenced the results of the study, and how could it

be used to improve the performance of novice software engineers.

VI. CONCLUSIONS AND FUTURE WORK

This paper presented an empirical study in which we verified to what extent novice software engineers can apply design patterns in the redesign of user interfaces of mobile applications. In their answers to a questionnaire of their experience applying the user interface design patterns, the novice software engineers indicated that they would apply the patterns again if given the chance and indicated specific activities such as how to select the problems to be corrected and patterns to address them. However, they indicated that the lack of experience and knowledge on how to apply the design patterns would make it difficult to do so. As improvement opportunities, we identified that document templates and a set of rules could be useful for guiding the redesign process, making it easier.

One of the limitations of our study was the small sample size employed and that the subjects were senior-level undergraduate students. However, even with the small sample, the results from this study allowed us to test the applicability of the design patterns for user interface design with positive results. Additionally, students who do not have experience in industry may have similar skills as novice software engineers in the industry [17]. Thus, the feedback provided by the subjects was useful for identifying aspects that made it difficult to apply the design patterns and improvement opportunities.

There could have also been a threat to the validity of our results in terms of the redesigned applications and their representativeness. Although two mobile applications were redesigned and evaluated, as there are many other categories of mobile applications (Games, Health, News, others), we cannot state that our results apply to all mobile applications. Nonetheless, *Manaus Bus* and *Manaus in Theaters* are two real applications which can resemble a real development and UX evaluation scenario, thus the qualitative data can provide useful information in the needs and improvements of the applicability of design patterns in order to facilitate their use by novice software engineers.

We also have to consider the subjectivity of the data classification as a threat to the validity of our results in the qualitative analysis, since it was performed by the first author of this paper. However, we used GT procedures in order to mitigate this threat, given that it requires the entire analysis to be grounded in the data collected. Additionally, the analysis process was performed along with another researcher, to encourage a better validation of the interpretations through the mutual agreement of the researchers.

Finally, one last limitation is the instrument and measures applied in this study for assessing the novice software engineers' opinion towards the use of design patterns. However, we believe that applying questionnaires was more suitable than applying interviews due to time constraints.

Given that each qualitative study provides evidences and hypotheses that can be later tested using quantitative methods, we intent to evaluate how including the improvement

opportunities identified in this study can facilitate the application process of design patterns by novice software engineers. Our intention is to provide means of evaluating UX features and facilitating the redesign process through the use of design patterns, so novice software engineers can improve the quality of their applications, improving their acceptance.

ACKNOWLEDGMENT

We thank scholarship granted by CNPq to the first author of this paper, the financial support granted by CAPES through PROAP program and the financial support granted by FAPEAM through processes numbers: 062.00600/2014; 062.00578/2014. Also, we thank all the students who participated in our study as novice software engineers and the users who tested the apps.

REFERENCES

- [1] J. Bargas-Avila, and K. Hombæk, "Old wine in new bottles or novel challenges: a critical analysis of empirical studies of user experience," Proc. SIGCHI Conference on Human Factors in Computing Systems, 2011, pp. 2689-2698.
- [2] M. Hassenzahl, and N. Tractinsky, "User experience-a research agenda," *Behaviour & information technology*, 25(2), 2006, pp. 91-97.
- [3] M. Ervasti, S. Dashti, J. Reilly, J. Bray, A. Bayen, and S. Glaser, "iShake: mobile phones as seismic sensors--user study findings," Proc. 10th international Conference on Mobile and Ubiquitous Multimedia, 2011, pp. 43-52.
- [4] M. Hassenzahl, "The thing and I: understanding the relationship between user and product," *Funology*, Springer Netherlands, 2005, pp. 31-42.
- [5] A. Seffah, "The evolution of design patterns in HCI: from pattern languages to pattern-oriented design," Proc. 1st International Workshop on Pattern-Driven Engineering of Interactive Computing Systems, 2010, pp. 4-9.
- [6] E. Chung, J. Hong, J. Lin, M. Prabaker, J. Landay, and A. Liu, "Development and evaluation of emerging design patterns for ubiquitous computing," Proc. 5th conference on Designing interactive systems: processes, practices, methods, and techniques, 2004, pp. 233-242.
- [7] R. Lanzilotti, C. Ardito, M. Costabile, and A. De Angeli, "Do patterns help novice evaluators? A comparative study," *International journal of human-computer studies*, 69(1), 2011, pp. 52-69.
- [8] K. Koukouletsos, B. Khazaei, A. Dearden, and D. Tseles, "Comparing patterns and guidelines in web design," Proc. 1st International Scientific Conference eRA, 2006.
- [9] C. Seaman, "Qualitative Methods," In: Shull et al. (eds.), *Guide to Advanced Empirical Software Engineering*, Chapter 2, Springer, 2008.
- [10] J. Tidwell, "Interaction Design Patterns," Conference on Pattern Languages of Programming, 1998. Available at: http://www.mit.edu/~jtidwell/interaction_patterns.html
- [11] M. Welie, "A Pattern Library for Interaction Design," 2011. Available at: <http://www.welie.com/patterns/index.html>.
- [12] E. Nilsson, "Design patterns for user interface for mobile applications," *Advances in Engineering Software*, 40(12), 2009, pp. 1318-1328.
- [13] Google. *Android Design Patterns*, 2016. <https://www.google.com/design/spec/patterns>
- [14] T. Neil, "Mobile design pattern gallery: UI patterns for smartphone apps," O'Reilly Media, 2014.
- [15] J. Nielsen, "Heuristic evaluation," *Usability Inspection Methods*, John Wiley & Sons, New York, 1994, pp. 25-62.
- [16] A. Strauss, and J. Corbin, "Basics of qualitative research," Newbury Park, CA: Sage. J., Vol. 15, 1990.
- [17] J. Carver, L. Jaccheri, S. Morasca, and F. Shull, "Issues in Using Students in Empirical Studies in Software Engineering Education," Proc. 9th International Symposium on Software Metrics, 2003, pp. 239 - 249.

Software Evolution by Correctness Enhancement *

Wided Ghardallou
University of Tunis El Manar
Tunis, Tunisia
wided.ghardallou@gmail.com

Nafi Diallo
NJIT
Newark, NJ 07102-1982 USA
ncd8@njit.edu

Ali Mili
NJIT
Newark, NJ 07102-1982 USA
mili@njit.edu

ABSTRACT

Relative correctness is the property of a program to be more-correct than another with respect to a specification; this property enables us to rank candidate programs in a partial ordering structure whose maximal elements are the correct programs. Whereas traditionally we think of program derivation as a process of successive correctness-preserving transformations (using refinement) starting from the specification, we argue that it is possible to derive programs by successive correctness-enhancing transformations (using relative correctness) starting from `abort`. One of the attributes of our approach is that it captures in the same mathematical model, not only the derivation of programs from scratch, but also most (if not all) of the activities that arise in software evolution. Given that most software is developed nowadays by evolving existing products rather than from scratch, any advance in the technology of program transformation by correctness enhancement stands to yield significant practical benefits.

Keywords

Program correctness, Relative correctness, Absolute correctness, software maintenance, software evolution, corrective maintenance, adaptive maintenance, software merger, software upgrade.

1. SOFTWARE EVOLUTION

Relative correctness is the property of a program to be more-correct than another with respect to a given specification. Intuitively, program P' is more-correct than program P with respect to specification R if and only if P' obeys R more often (for a larger set of inputs) than P , and violates R less egregiously (in fewer ways) than P . We came across relative correctness in our attempt to define what is a fault in a program, and what is fault removal [2, 13]. We have since explored the impact of relative correctness in software engineering [3], in software testing [5], and in software design [4]. In this paper, we build upon our results of [4] by showing that relative correctness can be used not only for program development from scratch, but also for various forms of program evolution. We argue, in fact, that virtually all software evolution is nothing but an effort to make

some program more-correct with respect to some specification. Given that today most software is developed, not from scratch, but rather by evolving existing software products, we feel that exploration of this avenue may yield substantial returns across software engineering practice. Our purpose, in this paper, is not to offer polished/ validated/ scalable solutions; rather, it is merely to highlight some of the opportunities that are opened by relative correctness in the field of software evolution.

All our definitions and results are formulated in terms of (binary) relations, hence we assume the reader familiar with elementary relational concepts, though we will introduce our own notations as we go. In section 2 we introduce a general framework in which we model relevant concepts such as specifications, program functions, program correctness, refinement, etc; and in section 3 we define relative correctness for deterministic and non-deterministic programs, then we review some relevant properties that we need in the remainder of the paper. In sections 4 to 7, we use relative correctness to model several aspects of software evolution, including: the derivation of a correct program from a specification; the derivation of a (not necessarily correct but sufficiently) reliable program from a specification; the merger of two programs; the upgrade of a program with a new feature; the removal of a fault from a program (corrective maintenance); and the transformation of a program to satisfy a new specification (adaptive maintenance).

2. A PROGRAMMING FRAMEWORK

When a program p manipulates variables x and y , say, of type X and Y , we let S be the cartesian product $X \times Y$, and we refer to S as the *space* of p and to an element of S as a *state* of p . We denote the X -component (resp. Y -component) of state s by $x(s)$ (resp. $y(s)$), and we may write simply x (resp. y) if no ambiguity arises; whatever decoration a state has (e.g. s' , or s'') is carried over to variable names (hence we write x' for $x(s')$ and x'' for $x(s'')$). Given a program p on space S , we let the *function* of p be the set of pairs of states (s, s') such that if execution of p starts in state s then it terminates in state s' ; we denote this function by upper case P and we may, by abuse of notation, refer to a program p and its function P interchangeably.

Given two relations R and R' on set S , we say that R' *refines* R (denoted by $R' \sqsupseteq R$ or $R \sqsubseteq R'$) if and only if $RL \cap R'L \cap (R \cup R') = R$, where L is the universal relation on S (i.e. $L = S \times S$) and the concatenation of two relations represents their product ($RR' = \{(s, s') \mid \exists s'' : (s, s'') \in R \wedge (s'', s') \in R'\}$). Note that RL can be writ-

*DOI reference number: 10.18293/SEKE2016-095

ten as $\{(s, s') | s \in \text{dom}(R)\}$; it represents the domain of R in relational form. A program p is said to be *correct* with respect to a specification (relation) R on S if and only if $P \sqsupseteq R$. This definition is equivalent to traditional definitions of total correctness [7], [8].

The refinement ordering has lattice-like properties, which we briefly present below:

- Two relations admit a least upper bound if and only if they satisfy the following condition, which we call the *consistency condition*: $RL \cap R'L = (R \cap R')L$.
- Whenever two relations R and R' do satisfy the consistency condition, their least upper bound (*join*) is defined by: $R \sqcup R' = \overline{R'L} \cap R \cup \overline{RL} \cap R' \cup (R \cap R')$.
- Two relations have a least upper bound if and only if they have an upper bound.

3. RELATIVE CORRECTNESS

3.1 Definition

Given a specification R and a program P , we refer to the domain of $(R \cap P)$ as the *competence domain* of program P with respect to specification R . When P is deterministic, the competence domain of P with respect to R is the set of initial states for which P behaves according to R ; when P is not deterministic (i.e. it may assign more than one final state for a given initial state), the competence domain of P is the set of initial states for which the set of images by P and the set of images by R overlap (i.e. have a non-empty intersection). We have the following definition, due to [2]:

DEFINITION 1. *Given two programs p and p' on space S and a specification R on S , we say that p' is more-correct than p with respect to R (written as: $P' \sqsupseteq_R P$ or $P \sqsubseteq_R P'$) if and only if $(R \cap P)L \subseteq (R \cap P')L \wedge (R \cap P)L \cap \overline{R} \cap P' \subseteq P$. We say that p' is strictly more-correct than p if one of the inclusions above is strict (\subset vs. \subseteq).*

The first clause provides that p' has a larger competence domain than p ; the second clause provides that on the competence domain of p , p' violates R in fewer ways than p (since any pair (s, s') such that s is in the competence domain of p and (s, s') is in P' and is not in R , is necessarily in P). In [2], we find that whenever P' is deterministic, the condition of relative correctness can be simplified as follows.

PROPOSITION 1. *Given two programs p and p' on space S such that p' is deterministic, and given a specification R on S , p' is more-correct than p with respect to R if and only if: $(R \cap P)L \subseteq (R \cap P')L$.*

In the remainder of this paper, we restrict our study to deterministic programs. Figure 1 shows an example of two deterministic programs p and p' and a specification R such that p' is more-correct than p ; the competence domains of p and p' are highlighted. Two observations are in order regarding this example: Note that p' is more correct than p with respect to R , but they are both incorrect; also note that even though p' is more-correct than p , it does not duplicate the correct behavior of p , rather it has its own correct behavior.

DEFINITION 2. *A fault in program p with respect to a specification R is any feature f (lexeme, expression, statement, set of statements, etc) that admits a substitute that would make the program strictly more-correct. A fault removal is a pair of features (f, f') such that f appears in p and program p' obtained by replacing f with f' in p is strictly more-correct than p .*

In the same way that refinement provides a formal model for program derivation, relative correctness, as defined herein, provides a formal model for fault removal; see Figure 3 (a). To give the reader some confidence in the soundness of our definition of relative correctness, we present the following properties, due to [13]:

- Relative correctness is reflexive and transitive, but not antisymmetric.
- Relative correctness culminates in (absolute) correctness, i.e. a correct program is more-correct than (or as correct as) any candidate program.
- Relative correctness logically implies (but is not equivalent to) higher reliability; relative correctness and higher reliability are not equivalent because the former is a logical property, whereas the latter is a stochastic property.
- The fourth property is the subject of the next proposition.

PROPOSITION 2. *Given two programs P and P' , P' refines P if and only if P' is more-correct than P with respect to any specification R . We write this as:*

$$P' \sqsupseteq P \Leftrightarrow (\forall R : P' \sqsupseteq_R P).$$

3.2 Relative Correctness and Refinement

Refinement is usually viewed as the touchstone of the derivation of provably correct programs; in such a derivation process, each transformation maps an artifact into a more-refined artifact. But Proposition 2 provides that p' refines p if and only if p' is more-correct than p with respect to *any* specification. If we are trying to derive a program that is correct with respect to a given specification R , then R and only R ought to be the focus of our derivation effort. In [4] we argue that it is possible to derive a correct program from a specification by stepwise correctness enhancing transformations using relative correctness, rather than by stepwise correctness preserving transformations using refinement. In this section, we briefly discuss the difference between a correctness-preserving transformation and a correctness-enhancing transformation.

As an illustrative example, we consider a space S defined by two integer variables x and y and we let R be the following specification on S : $R = \{(s, s') | x' = x + y\}$. We consider the following candidate programs:

$$\begin{aligned} p_0: & \{\text{while } (y \neq 0) \{y=y-1; x=x+1;\}\}, \\ P_0 & = \{(s, s') | y \geq 0 \wedge x' = x + y \wedge y' = 0\}. \end{aligned}$$

$$p_1: \{x=x+y;\}. P_1 = \{(s, s') | x' = x + y \wedge y' = y\}.$$

$$p_2: \{x=x+y; y=0;\}, P_2 = \{(s, s') | x' = x + y \wedge y' = 0\}.$$

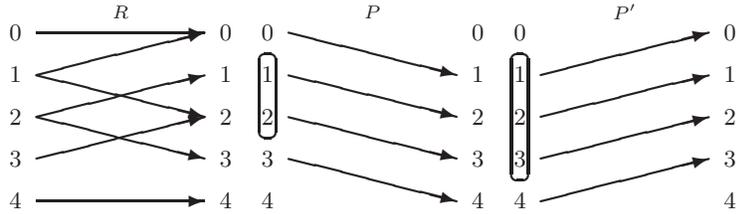


Figure 1: Enhancing correctness without duplicating behavior: $P' \sqsupseteq_R P$

The reader can check (by referring to the definition of refinement given in section 2 and by Proposition 1) that p1 is more-correct than p0 but does not refine it, whereas p2 refines p0 hence (according to Proposition 2) is more-correct than p0. We have a simple explanation for this observation: if we consider the functional attributes of p0, we can distinguish between two sources of information:

- Functional attributes that are mandated by the specification, such as the clause $\{x' = x + y\}$ in P_0 .
- Functional attributes that are not mandated by the specification, but stem from design decisions, such as the clause $\{y' = 0\}$ in P_0 .

In order for a program to refine P_0 , it has to preserve all the functional attributes of P_0 , regardless of their origin (specification or design); this is the case for p2. But in order for a program to be more-correct than P_0 , it suffices for it to preserve the functional attributes that are mandated by R ; it may, in the process, override / alter functional attributes that stem from design decisions; this is the case for program p1 which overrides $\{y' = 0\}$ with $\{y' = y\}$. Not surprisingly, p1 is simpler than p2, because it is subject to a less stringent condition.

4. PROGRAM DERIVATION

4.1 Correct Programs

To derive a correct program for a specification R , we have to find an artifact that has two properties: it is a (executable) program; and it is correct with respect to R . To do this in a stepwise manner, we can either preserve correctness until we achieve executability (this is the traditional refinement-based process), or we can maintain executability until we achieve correctness (as we advocate in [4]). These two processes being iterative, we can characterize them by their initial state, their invariant assertion, their variant function, and their exit condition, as shown in Figure 2. See also Figure 3 (b).

4.2 Reliable Programs

In many applications, correctness is unnecessary, and the cost of achieving correctness (as opposed to sufficient reliability) may be unjustified by the stakes attached to (sufficiently infrequent) program failure. In such cases, it may be sufficient to produce a reliable program, rather than a correct program. Thankfully, the derivation model presented herein encompasses the derivation of reliable programs as well as the derivation of correct programs: as we remember from section 3, relative correctness logically implies higher

Attribute	Refinement Based	Based on Relative Correctness
Initialization	$a = R$	$a = \text{abort}$
Invariant Assertion	a is correct	a is a program
Variant Function	a increasingly concrete (program-like)	a increasingly correct
Exit test	when a is a program	when a is correct

Figure 2: Iterative Paradigms of Programming

reliability; hence the successive programs generated by this process are increasingly reliable; so that the derivation of a reliable program proceeds in the same way as the derivation of a correct program, except that it terminates as soon as the reliability reaches or exceeds the required threshold. Given that correctness is the culmination of reliability, it is only fitting that the derivation of a correct program be the culmination of the derivation of a reliable program. See Figures 3 (b) and (c).

5. PROGRAM MERGER

We consider a specification R and two candidate programs P_1 and P_2 (i.e. programs that are written to satisfy R –they may or may not satisfy it in fact), each of which fulfills the requirements of R to some limited extent, but not necessarily to the full extent. We are interested to merge programs P_1 and P_2 into a program that fulfills the requirements of R to the extent that P_1 fulfills them, and to the extent that P_2 fulfills them. We submit the following definition.

DEFINITION 3. *Given a specification R and two candidate programs P_1 and P_2 , a merger of P_1 and P_2 with respect to R is any program P' that is more-correct than P_1 and more-correct than P_2 with respect to R .*

We mandate that a merger program be merely more-correct than programs P_1 and P_2 , rather than to refine them, for the following reasons:

- *Refinement is Unnecessary.* When we resolve to refine a program, we commit to refine all its functional attributes, those that are mandated by the specification as well as those that stem from design decisions. But we have no reason to preserve design decisions of P_1 and P_2 that do not advance the cause of relative correctness.

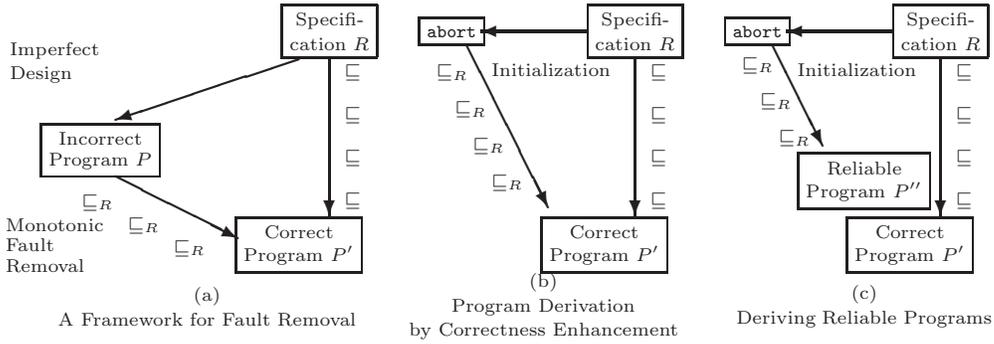


Figure 3: Alternative Program Evolution Paradigms

- *Relative Correctness is Sufficient.* If program P' is more-correct than P_1 and P_2 with respect to specification R , then it delivers all the specification-mandated behavior of P_1 and all the specification-mandated behavior of P_2 .
- *Refinement may be Impossible.* Not only is it unnecessary to refine the design-related information of P_1 and P_2 , it may actually be impossible: whereas the specification-mandated information of P_1 and P_2 is bounded by R , hence (according to section 2) can be combined by the least upper bound operation, the design-related information of P_1 and P_2 may be incompatible, hence cannot be combined.

We consider the space S defined by three variables x , y and z of type integer, and we let R be the following specification: $R = \{(s, s') | x' = x + y \wedge z' \geq z + 2\}$. Let p_1 and p_2 be the following candidate programs for specification R :

p1: $\{z=z+2; \text{while } (y!=0) \{y=y-1; x=x+1;\}\}$
p2: $\{z=z+3; \text{while } (y!=0) \{y=y+1; x=x-1;\}\}$

The functions of these programs are, respectively:

$P_1 = \{(s, s') | y \geq 0 \wedge x' = x + y \wedge y' = 0 \wedge z' = z + 2\}$
 $P_2 = \{(s, s') | y \leq 0 \wedge x' = x + y \wedge y' = 0 \wedge z' = z + 3\}$.

Indeed, the first program terminates only for initial y greater than or equal to zero, and when it terminates, the final value of x contains $x + y$, the final value of y is zero, and z is incremented by 2. As for the second program, it terminates only for non-positive y , and when it does terminate, the final value of y is zero, z is increased by 3 and x contains $x + y$. So that each program does some of what R asks, but neither is correct. A merger of these two programs is any program P' that is more-correct than P_1 and more-correct than P_2 with respect to R . The systematic derivation of the merger of two programs is beyond the scope of this paper; we content ourselves with presenting a candidate program then showing that it satisfies the definition of a merger. We propose:

p': $\{z=z+4;$
 if $(y>0) \{\text{while } (y!=0) \{y=y-1; x=x+1;\}\}$
 else $\{\text{while } (y!=0) \{y=y+1; x=x-1;\}\}$

As far as x and y are concerned, this program imitates the behavior of P_1 for non-negative values of y , and the behavior of P_2 for non-positive values of y ; as far as z is concerned, this program overrides the behavior of both P_1 and P_2 and increments z by 4. We argue that this program is more-correct than P_1 and more-correct than P_2 with respect to R . The function of this program is:

$$P' = \{(s, s') | x' = x + y \wedge y' = 0 \wedge z' = z + 4\}.$$

Space restrictions preclude us from showing details, but it is easy to verify that the competence domain of P' ($(R \cap P)L$) is equal to L , hence P' is more-correct than P_1 and P_2 . Note that while we found a program that is more-correct than P_1 and P_2 , we could not find a program that refines P_1 and P_2 . Indeed we can easily check that P_1 and P_2 do not satisfy the consistency condition, hence they admit no joint refinement. Indeed, no program can simultaneously increase z by 2 (to refine P_1) and by 3 (to refine P_2). This discrepancy between what P_1 does and what P_2 does precludes P_1 and P_2 from having a joint refinement, but does not preclude them from having a program P' that is more-correct than them. The reason is: the statements $\{z=z+2\}$ (in P_1) and $\{z=z+3\}$ (in P_2) are not mandated by the specification (which only requires $\{z' \geq z + 2\}$) but stem instead from arbitrary design decisions; hence both can be overridden by the merger program P' . The difference between refinement and relative correctness is that the former attempts to refine all the behavior of a program, regardless of its source, whereas the latter only refines the behavior that is mandated by the specification. As we see in this simple example, refining all the behavior of P_1 and all the behavior of P_2 is not only unnecessary, it is actually impossible. See Figure 4 (a), where R_1 and R_2 represent the specification-mandated behavior of P_1 and P_2 (we have explicit formulae for these, but their study is beyond the scope of this paper).

6. PROGRAM UPGRADE

We are given a specification R and a candidate program P , and we are interested to augment program P with a new feature that is specified by some relation Q . Typically, P may be a large, complex, comprehensive application that delivers a wide range of services, and Q is a punctual additional function or service that we want to incorporate into P (for example, P is a sprawling corporate data processing application, and Q specifies an additional report to be delivered, or an additional output screen, or an additional statistic on corporate transactions, etc). In transforming P into P' , we have every expectation that P' refines Q , because Q is a fairly simple requirement and because it is the main goal of the operation. But we have no expectation that P' refine P , because the implementation of Q may require that some of the behavior of P be altered. Nor do we expect that P' refines R , because in fact we are not even sure P refines R (P is typically incorrect, i.e. it fails to correctly deliver all the required services in all circumstances).

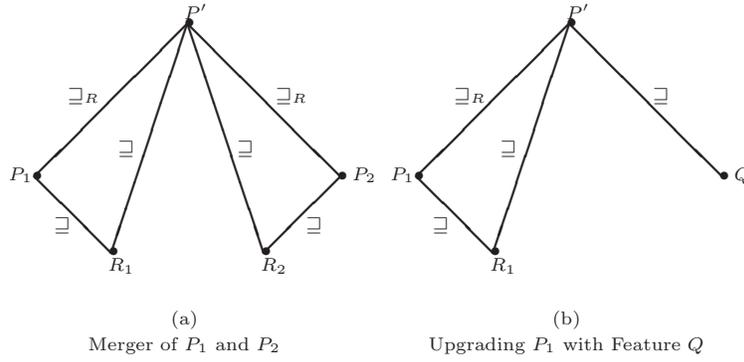


Figure 4: Merger and Upgrade

While we do not expect P' to refine P nor R , we most certainly expect P' to be more-correct than P with respect to R ; in other words, we do not want that in the process of adding feature Q to P , we degrade the correctness of P with respect to R .

DEFINITION 4. *Given a specification R and a candidate program P , and given a feature Q that we want to add to P , an upgrade P' of P with feature Q is any program that refines Q and is more-correct than P with respect to R .*

Given a specification R on space S defined by integer variables x and y , $R = \{(s, s') | x' = x + y\}$ and given the following candidate program,

p_1 : $\{x=x+10; \text{while } (y!=10) \{y=y-1; x=x+1;\}\}$

we consider the problem of upgrading program P_1 with feature Q defined by: $Q = \{(s, s') | y > 0 \wedge y' = 0\}$. The function of program p_1 is:

$P_1 = \{(s, s') | y \geq 10 \wedge x' = x + y \wedge y' = 10\}$.

Note that P_1 and Q do not satisfy the consistency condition, since P_1 sets y to 10 while Q mandates that we set it to 0 (for positive values of y). Therefore it is impossible to fulfill requirement Q without altering the behavior of P_1 . Fortunately, the feature of P_1 that precludes us from refining Q , namely the clause $y' = 10$, is not a specification-mandated requirement, but stems instead from the specific design of P_1 . Hence while it is impossible for the upgrade program P' to refine P , it is not impossible for P' to be more-correct than P with respect to R . We consider the following program:

p' : $\{\text{while } (y!=0) \{y=y-1; x=x+1;\}\}$

The function of this program is:

$P' = \{(s, s') | y \geq 0 \wedge x' = x + y \wedge y' = 0\}$.

While space limitation preclude us from showing detailed calculations, it is easy to check that P' does refine Q . On the other hand, we can easily check that the competence domain of P_1 is $\{(s, s') | y \geq 10\}$ whereas the competence domain of P' is $\{(s, s') | y \geq 0\}$. Hence P' is more-correct than P_1 with respect to R . While it is not possible to satisfy Q while preserving all the behavior of P_1 , it is possible, and sufficient, to satisfy Q while enhancing the correctness of P_1 ; this is what P' does. See Figure 4 (b).

7. SOFTWARE MAINTENANCE

7.1 Corrective Maintenance

We argue in this section that corrective maintenance is nothing but an instance of program transformation by relative correctness: in fact it is merely a step in the process we have outlined for program derivation by correctness enhancement; it starts at the current program (rather than abort) and it ends a step later (rather than necessarily at a correct program). See Figure 5. As an illustration, we consider the following program on the space S defined by its variable declarations (due to [6], with some modifications):

```
p: #include <iostream> ... .. // line 1
main {(char q[]); int let, dig, other, i, l; char c; // 2
    i=0; let=0; dig=0; other=0; l=strlen(q); // 3
    while (i<l) { // 4
        c = q[i]; // 5
        if ('A'<=c && 'Z'>c) let+=2; // 6
        else // 7
            if ('a'<=c && 'z'>=c) let+=1; // 8
        else // 9
            if ('0'<=c && '9'>=c) dig+=1; // 10
            else // 11
                other+=1; // 12
        i++; // 13
    }
```

We define the following sets: $\alpha_A = \{ 'A' \dots 'Z' \}$. $\alpha_a = \{ 'a' \dots 'z' \}$. $\nu = \{ '0' \dots '9' \}$. $\sigma = \{ ' ' , ' - ' , ' = ' , \dots , ' / ' \}$, the set of all the ascii symbols. We let $list\langle T \rangle$ denote the set of lists of elements of type T , and we let $\#_A$, $\#_a$, $\#_\nu$ and $\#_\sigma$ be the functions that to each list l assign (respectively) the number of upper case alphabetic characters, lower case alphabetic characters, numeric digits, and symbols. We consider the following specification on S :

$R = \{(s, s') | q \in list\langle \alpha_A \cup \alpha_a \cup \nu \cup \sigma \rangle \wedge$
 $let' = \#_a(q) + \#_A(q) \wedge dig' = \#_\nu(q) \wedge other' = \#_\sigma(q)\}$.

The competence domain of P is:

$(R \cap P)L = \{(s, s') | q \in list\langle \alpha_a \cup \nu \cup \sigma \rangle\}$.

This is different from the domain of R , which is

$RL = \{(s, s') | q \in list\langle \alpha_A \cup \alpha_a \cup \nu \cup \sigma \rangle\}$,

hence P is not correct with respect to R . If we let P' be the program obtained from P by changing $\{let+=2\}$ into $\{let+=1\}$, we find:

$(R \cap P')L = \{(s, s') | q \in list\langle (\alpha_A \setminus \{ 'Z' \}) \cup \alpha_a \cup \nu \cup \sigma \rangle\}$.

Clearly, $(R \cap P')L \supset (R \cap P)L$. Hence statement $\{let+=2\}$ is a fault in P with respect to specification R and the substitution of $\{let+=2\}$ by $\{let+=1\}$ is a fault removal in P with respect to R .

7.2 Adaptive Maintenance

Adaptive maintenance consists in taking a program P which was originally developed to satisfy some specification

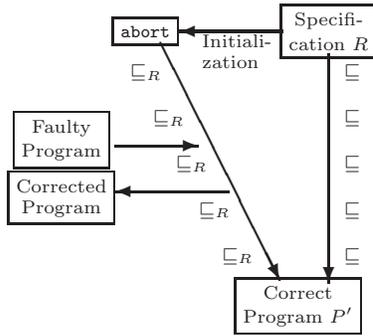


Figure 5: Corrective Maintenance

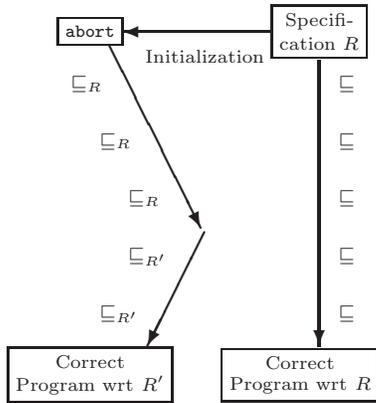


Figure 6: Adaptive Maintenance

R and changing it to make it satisfy some new specification R' . We view this as simply trying to make P more-correct with respect to R' than it is in its current form. Clearly, one does this if one believes that P is close enough to satisfy R' that it is more economical to evolve P than to start from `abort`. Be that as it may, we argue that adaptive maintenance is again a process of making a program more-correct with respect to a given specification. See Figure 6.

8. CONCLUSION

In this paper, we consider the concept of relative correctness (due to [13]), and show that it pervades software evolution, and is potentially more flexible, without being less effective, than refinement-based program transformations. In particular, we find that this concept can provide a formal model for a wide range of software evolution activities, including software design, corrective maintenance, adaptive maintenance, software upgrade, program merger, etc. As a consequence, we argue that by evolving a technology of program transformation with relative correctness, we stand to enhance a wide range of software engineering activities.

Other authors have introduced similar-sounding but distinct notions of relative correctness [1, 9–12, 14]. Their work differs from ours in terms of its specification format (executable assertions vs. input/output relations), its program semantics (execution traces vs. program functions), its definition of correctness (successful assertions vs. refinement),

its definition of relative correctness (more successful/ less unsuccessful assertions, vs. larger competence domains), and its goals (fault diagnosis/ program repair vs. software evolution).

9. REFERENCES

- [1] Borzoo Bonakdarpour, Ali Ebneenasir, and Sandeep S. Kulkarni. Complexity results in revising unity programs. *ACM Transactions on Autonomous and Adaptive Systems*, 4(1), January 2009.
- [2] Jules Desharnais, Nafi Diallo, Wided Ghardallou, Marcelo Frias, Ali Jaoua, and Ali Mili. Mathematics for relative correctness. In *Relational and Algebraic Methods in Computer Science, 2015*, pages 191–208, Lisbon, Portugal, September 2015.
- [3] Nafi Diallo, Wided Ghardallou, and Ali Mili. Correctness and relative correctness. In *Proceedings, 37th International Conference on Software Engineering*, Firenze, Italy, May 20–22 2015.
- [4] Nafi Diallo, Wided Ghardallou, and Ali Mili. Program derivation by correctness enhancements. In *Refinement 2015*, Oslo, Norway, June 2015.
- [5] Wided Ghardallou, Nafi Diallo, Ali Mili, and Marcelo Frias. Debugging without testing. In *Proceedings, International Conference on Software Testing*, Chicago, IL, April 2016.
- [6] A. Gonzalez-Sanchez, R. Abreu, H-G. Gross, and A.J.C. van Gemund. Prioritizing tests for fault localization through ambiguity group reduction. In *proceedings, Automated Software Engineering*, Lawrence, KS, 2011.
- [7] D. Gries. *The Science of programming*. Springer Verlag, 1981.
- [8] E.C.R. Hehner. *A Practical Theory of Programming*. Prentice Hall, 1992.
- [9] Barbara Jobstmann, Andreas Griesmayer, and Roderick Bloem. Program repair as a game. In *Computer Aided Verification*, number 3576 in LNCS, pages 226–238, 2005.
- [10] Shuvendu K. Lahiri, Kenneth L. McMillan, Rahul Sharma, and Chris Hawblitzel. Differential assertion checking. In *Proceedings, ESEC/ SIGSOFT FSE*, pages 345–455, 2013.
- [11] Francesco Logozzo and Thomas Ball. Modular and verified automatic program repair. In *Proceedings, OOPSLA*, pages 133–146, 2012.
- [12] Francesco Logozzo, Shuvendu Lahiri, Manual Faehndrich, and Sam Blackshear. Verification modulo versions: Towards usable verification. In *Proceedings, PLDI*, page 32, 2014.
- [13] Ali Mili, Marcelo Frias, and Ali Jaoua. On faults and faulty programs. In Peter Hoefner, Peter Jipsen, Wolfram Kahl, and Martin Eric Mueller, editors, *Proceedings, RAMICS: 14th International Conference on Relational and Algebraic Methods in Computer Science*, volume 8428 of *Lecture Notes in Computer Science*, pages 191–207, Marienstatt, Germany, April 28–May 1st 2014. Springer.
- [14] Hoang Duong Thien Nguyen, DaWei Qi, Abhik Roychoudhury, and Satish Chandra. Semfix: Program repair via semantic analysis. In *Proceedings, ICSE*, pages 772–781, 2013.

Risk Management Analysis in Software Projects which Use the Scrum Framework

Breno Gontijo Tavares¹ Carlos Eduardo Sanches da Silva¹ Adler Diniz de Souza²

¹Instituto de Engenharia de Produção e Gestão
Universidade Federal de Itajubá, UNIFEI
Itajubá, Brazil

²Instituto de Matemática e Computação
Universidade Federal de Itajubá, UNIFEI
Itajubá, Brazil

Abstract

Studies show one of the reasons for the failure of the software projects is the absence of Risk Management or its improper application. The adoption of Scrum Framework on software projects is increasing. However, the Scrum does not have specific Risk Management activities. In this scenario, this paper presents the results of a survey applied using qualitative approach, in order to analyze how Risk Management is carried out in software projects, which use Scrum. The research method adopted was the case study and the research instrument for data collection was developed based on scientific articles and the application of structured interviews. As a result, this paper presents Risk Management practices which achieved greater and lower agreement among respondents and literature. It was found that Risk Management must be applied continuously in a feedback loop. Furthermore, Scrum projects must not have a high formal planning level, even for the high risk ones. This result does not converge to the literature. The research verified that the Risk Management in Scrum projects is performed differently from its application on traditional methodologies. The framework has native resources, but the use of the classic Risk Management processes must be incorporated and adapted to Scrum.

Keywords: *Risk Management; Scrum; Software; Case Study.*

I. INTRODUCTION

Software projects are complex enterprises in any context and they are particularly susceptible to failure [1]. Most of these projects run over budget are terminated prematurely or fall far short of meeting user expectations and business functionalities [2].

In this scenario, software development industry has been using agile methodologies to manage projects instead of using traditional methodologies [3, 4]. This is because traditional methodologies are commonly considered heavy, unlike agile methodologies, which aim to provide light approaches to projects [5].

The Scrum framework is the most used agile methodology in Software Project Management [6, 7, 8], which provides a set of good practices aimed at fast delivery of value to the customer.

However, Risk Management, which can reduce uncertainty and increase the chances of success in software projects [9, 10, 11, 12], is conducted in an implicitly way in projects which use agile methodologies [13, 14, 15]. Furthermore, risk management

in agile projects needs to be improved without threatening the agility of projects [16].

Nyfjord and Kajko-Mattsson [15] have done a comparative analysis between the traditional and agile risk management approaches. The authors assert that agile methodologies do not provide any risk management taxonomy and they suggest that agile methodologies should learn to integrate traditional risk management practices in order to ensure an effective risk management.

Furthermore, the literature about Risk Management applied in software development projects which use agile methodologies is scarce [17, 18]. There are few articles describing the Risk Management application with agile methodologies. However, these articles do not emphasize the process of how a team establishes, prioritizes and takes action about risks [19].

Therefore, the aim of this work is to answer the following research question: How Risk Management is performed in projects that use Scrum framework?

From this survey question, this paper aims to analyze the Risk Management in Scrum projects and establishes the following specific objectives:

- Identify applied Risk Management practices in software development;
- Conduct a case study in software projects that use the Scrum, in order to analyze the respondents agreement with Risk Management practices identified in scientific literature.

This paper is organized as follows: Section 1 presents the objectives and research contributions; Section 2 performs the literature review about Risk Management in Scrum; Section 3 presents the research classification, the case study design and research protocol; Section 4 presents the data collection and its results, and finally, Section 5 presents the discussions, conclusions and suggestion for further researches.

II. RISK MANAGEMENT IN THE SCRUM FRAMEWORK

Scrum life cycle provides the monitoring of the product that is being developed and identifies impediments. Some authors define impediment as a project risk [20, 21], while other authors state that there are differences between these two concepts [22, 23, 24].

This paper uses the impediment definition proposed by Jakobsen and Johnson [22], as a problem that has already occurred and it is affecting project progress and distinguishing risk and impediment definitions.

In this context, when Risk Management is used in Scrum projects, it enables the prevention of impediments,

implementing proactive actions to inhibit project risks to become further impediments [22].

According to [25], Scrum framework uses an iterative and incremental approach to optimize predictability and risk control. The authors state that the use of Sprints also supports Risk Management, since it limits risk to one calendar month of cost [25].

However, other authors believe that the Scrum and agile methodologies in general do not suggest specific activities to perform Risk Management [13, 14, 15], and this management in Scrum is not as good as traditional methodologies [26]. According to [26], the Scrum serves only for risks identification practice, however, it does not offer ways to analyze and manage them.

Despite the importance of Risk Management to project success, few scientific studies were identified in a survey conducted in the three following databases: Web of Science, Scielo and Google Scholar. The words “Scrum”, “Project”, “Risk” and “Management” were searched in articles titles without returning any results.

III. METHODOLOGY

The research method used was a case study (Yin, 2009), for the purpose of exploration. The methods used for data collection were the questionnaire and semi-structured interviews.

There were selected software projects that used the Scrum Framework. One of the selection criteria was the relevance of these projects in terms of invested effort, which was established a minimum of 1500 hours for each project.

As selection result, there were identified 10 software projects that met selection criteria, 2 of them under development and 8 finished projects. Five projects were developed for medium-sized companies while the other five were developed for large-sized companies. This classification was based on the European Commission [27] proposal, which establishes annual revenue greater than 50 million euros for large-sized companies, and annual revenue between 10 and 50 million euros for medium-sized companies.

Aiming to know these projects characteristics, their classification was performed according to diamond model, proposed by [28]. The model, also known as NTCP, provides four dimensions for project classification: Novelty, Technology Complexity and Pace. This Model is considered to be a suitable framework for project classification [29]. Other scientific researches used NTCP model for project classification [30, 31, 32].

Scrum Masters of these projects responded to a structured questionnaire applied through an on-site interview, aiming to classify each project according to NTCP model. The classification was validated by the senior managers of these projects through interviews.

Most projects were classified as “Platform”, “Medium-Tech”, “System” and “Fast/Competitive”. These classifications allow the identification of one of this research limitations, because applying the same research protocol in a different environment may present other results.

These Scrum Masters attended to another on-site interview to identify the adherence of the ten projects to the roles, events, artifacts and rules of Scrum. Assessing adherence of these

projects to Scrum was performed because the possibility of them do not implement Scrum in its entirety.

This analysis was performed following the 4 steps:

1. Identifying Scrum practices: this step aimed the lifting of the main features of Scrum according to the Scrum Guide book [25];
2. Preparing the questionnaire: the questionnaire was designed with seventeen closed questions, using a Likert scale of six points and classified according to the chapters of the Scrum Guide.
3. Performing the interviews: the Scrum Masters who worked on the ten projects were individually interviewed where the structured questionnaire was applied.
4. Analysing the results: the analysis revealed the smaller and larger adhesions, beyond the average adherence of each project to Scrum.

The Table 1 presents the questionnaire applied to identify the adherence of the then projects to the Scrum framework.

TABLE 1. QUESTIONNAIRE TO IDENTIFY THE ADHERENCE LEVEL TO SCRUM
THE COMPLETE TABLE CAN BE SEEN AT THE ADDRESS

<https://goo.gl/Mh1Cem>

Scrum Practices		Questions		Classification
Nº	Description	Nº	Description	
1	Scrum users must frequently inspect Scrum artifacts and progress toward a Sprint Goal to detect undesirable variances [25]	1	The team, frequently, inspect Scrum artifacts and progress toward Sprint Goal to detect undesirable variances.	Scrum Theory
11	When a Product Backlog item or an Increment is described as “Done”, everyone must understand what “Done” means [25].	17	When a Product Backlog item or an Increment is described as “Done”, everyone must understand what “Done” means.	Artifact Transparency

The results of the adherence study of the ten projects presents that three projects had no adherence to Scrum in at least one of the questions in the questionnaire. The project that received the highest average adherence to Scrum was 89.41% while the project who got the lowest value was 43.53%.

The differences in results between the projects must be justified considering the different context of each project, where for each one has defined a management scope to meet the project specific needs. This setting allowed the choice of what artifacts, events, and rules of Scrum that would be used in the project life cycle.

To develop the research protocol, a survey was conducted in the three databases presented in the Table 2.

TABLE 2 – DATABASES USED TO DEVELOP THE RESEARCH PROTOCOL

Database	Description
Web of Science	The Web of Science is the largest repository of scientific papers worldwide [33, 34].
Scielo	The Scielo database represents one of the major scholarly communication programs in the developing countries and is a pioneer in the adoption of Open Access [35].
Google Scholar	Google Scholar is a free web-based database which indexes literature in a wide variety of formats [36] and has steadily grown in importance in the academic library community [37].

The following terms were searched in the articles titles:

- (1) Software AND Project AND Risk AND Management
- (2) Agile AND Project AND Risk AND Management
- (3) Scrum AND Project AND Risk AND Management

All these search terms were combined by using the Boolean “OR” operator, which entails that an article only had to include any one of the terms to be retrieved. That is, we searched: 1 OR 2 OR 3 and we identified 138 articles.

This research considered Impact Factor (IF) from the Journal Citation Reports (JCR) as another article selection criteria, aiming to increase the Risk Management practices reliability. From the 138 identified articles initially, only 11 possess IF from JCR.

The 11 articles were deeply analyzed aiming to gather the Risk Management practices. There were 3 of the 11 analyzed articles that did not contain any of those practices and so could not be used in the research protocol development.

It is important to mention that the considered articles are not specifically about Scrum projects. However, they reference Risk Management applied to software projects. There was not identified any article that specifically analyzes Risk Management in this framework and that attended the selection criteria established for the research protocol.

The 8 articles were deeply analyzed and 35 Risk Management practices were identified considering only software projects. The identified practices were used to generate 40 questions. The questions were developed using a 6-point Likert scale since it does not allow the respondents to choose a central point, which can be considered as neutral or without opinion [38].

The questionnaire was submitted to two pilot tests in two different companies that develop software projects and use the Scrum framework for more than 4 years. Considering the total sum of the modifications from the two tests, there were 31 modifications on 40 questions or 77,5% of the questionnaire.

The Table 3 presents the practices and their respective Risk Management questions, after the application of the pilot tests.

TABLE 3 - RISK MANAGEMENT PRACTICES AND QUESTIONS
THE COMPLETE TABLE CAN BE SEEN AT THE ADDRESS <https://goo.gl/4JFTIM>

Risk Management Practices		Questions	
Nº	Description	Nº	Description
1	Senior management holds the key to establishing an organization that encourages 'functional' Risk Management behavior [39].	1	Senior management holds the key to establishing an organization that encourages Risk Management behavior.
35	Cannot be stated that formal Risk Management procedures produce better results than internal methods (ad hoc internally developed procedures) [12].	40	Cannot be stated that formal Risk Management procedures produce better results than internal methods (ad hoc internally developed procedures).

Beyond the research protocol 40 questions, there were developed 7 questions of external validation and 4 questions to supplement the questionnaire.

IV. RESULTS ANALYSIS

The field research was performed through on-site interviews, providing a better communication between the interviewer and respondents. Interviews from 40 to 60 minutes were applied for the 21 members of the selected projects. This population represented 100% of the employees that work on the project and that still worked on the organization.

The results of the external validation are presented by the Table 4.

TABLE 4 - EXTERNAL VALIDATION RESULTS

Questions	Results
Education level	<ul style="list-style-type: none"> • 66,67% = Specialization • 33,33% = Graduation • 0% = Technician, High School, Incomplete Graduation, Master and Doctor Degree
Which functions have you performed?	<ul style="list-style-type: none"> • 90,48% = Developer • 47,62% = Scrum Master • 19,05% = Owner
How long is your experience in software projects development?	<ul style="list-style-type: none"> • 80,95% = Beyond 4 years • 9,52% = From 1 to 2 years • 9,52% = From 2 to 4 years • 0% = Less than 1 year
How long is your experience in Scrum?	<ul style="list-style-type: none"> • 42,86% = From 2 to 4 years • 33,33% = From 1 to 2 years • 14,29% = Less than 1 years • 9,52% = Beyond 4 years

The interview identified that some of the interviewed professionals possess certifications in the areas of Project Management, Scrum and Risk Management, as follows on Table 5.

TABLE 5 – PROFESSIONAL CERTIFICATIONS

Certification	Interviewees
PMP (Project Management Professional)	4
PSM (Professional Scrum Master)	2
PMI-RMP (Risk Management Professional)	1

It is important to mention that the interviewer possess the three certifications presented on the Table 5.

The external validation was ensured through the profile of the interviewed professionals that work or worked in one of the ten selected projects.

Aiming to obtain the internal consistency of the questionnaire, the Cronbach Alpha [40], Theta coefficient (Θ) [41] and the Omega Indicator (Ω) [42] were calculated. The values obtained were, respectively, 0.8302, 0.8320 and 0.8875, which are within acceptable values [40, 41, 42].

In order to obtain a deeper analysis for the questions results, Table 6 was generated through the software Minitab 17®, performing the descriptive statistics for the Risk Management practices.

TABLE 6 – THE QUESTIONNAIRE AND DESCRIPTIVE STATISTICS
 THE COMPLETE TABLE CAN BE SEEN AT THE ADDRESS <https://goo.gl/6t1ZII>

Nº	Practices	Questions	Mean	S.E. Mean	Standard deviation	Minimum	Median	Maximum
32	Risk management should be performed continuously in a feedback loop so that problematic situations can be dynamically detected and adjusted [43].	37 Risk Management should be performed continuously in a feedback loop so that problematic situations can be dynamically detected and adjusted.	4,762	0,118	0,539	3	5	5
19	Since every project element may be related to uncertainty events that cause prejudice to a software project, development teams must reuse knowledge about risks that should occur in applications developed with these elements [44].	21 The development teams must reuse knowledge about risks (lessons learned).	4,619	0,109	0,498	4	5	5
16	Software development projects require high levels of Formal Planning for high Risk Exposure projects and low levels of Formal Planning for low Risk Exposure projects [45].	18 Software development projects require high levels of Formal Planning for high Risk Exposure projects and low levels of Formal Planning for low Risk Exposure projects.	1,667	0,261	1,197	0	2	3
30	After the initial round of risk identification, risk management tended to be relegated to the project manager, who often did little more than informally update the risk register before each steering committee meeting [1].	32 After the initial round of risk identification, Risk Management tended to be relegated to the SCRUM MASTER.	1,619	0,253	1,161	0	1	3

As can be observed in the Table 6, the practice with highest average is related to continuously accomplishing the Risk Management in a feedback loop, where Risk Management must be updated in each iteration with new project data, and then generate new estimates [41]. This means that the respondents had higher agreement with this practice when it is about a Scrum project rather than other identified Risk Management practices. This result converges to the Scrum life cycle that possess ceremonies of which the focus is to obtain a constant feedback [46, 47]. The authors [48] complements that the continuous feedback is the key factor for projects success.

The practice with the second highest average refers to the reuse of risks knowledge or the lessons learned. This result shows the respondents believe that this knowledge should be used in Scrum projects. The lessons learned are commonly used within Scrum life cycle and are mainly identified and discussed in retrospective meetings [49].

On the other hand, the research identified that the Risk Management practice with the lowest average is related to the Risk Management tend to be relegated to the Scrum Master after the initial round of risk identification. It shows that the respondents in general agree less with this practice, in other words, that they believe Risk Management occurs within other project occasions. The risks identification on Scrum is iterative, occurring during daily meetings [20] or any other Scrum meeting [50].

The practice with the second lowest average refers to the fact that high-risk projects demand high formal planning levels and low-risk projects demand low formal planning levels. The respondents in general did not agree with this practice, justifying the Scrum processes must be agile, even in high risk projects, and a high level formal planning impacts negatively on agility. This opinion is different than that found in the literature. For example, [51] defends that agile methodologies must be used considering the project context for adapting the processes.

There were also identified results that do not converge to the Scrum literature. For example, the practice related to the Scrum Master unavailability to implement a formal Risk Management

process. The respondents in general believe that, in practice, the Scrum Master is not concerned with the implementation of formal processes and his role is focused on assisting the development team.

On the other hand, some authors claim that the Scrum Master is responsible for managing Scrum processes [52] and other processes used for developing the software [53]. The authors [54] claim that the Scrum Master is responsible for the implementation of the project risks plan.

During the same interview were applied four additional questions, whose aim was to complement the Risk Management questionnaire in Scrum.

All respondents believe that Scrum has no specific activities for the practice of risk management, but that the ceremonies of the framework allow the treatment of risks, converging with the opinion of other authors [13, 14, 15].

For 76.2% of the respondents, the Risk Management in Scrum is different from that practiced in traditional methodologies. The respondents mentioned that there is no defined processes for risk management in the Scrum framework, while traditional methodologies have bureaucratic processes to implement this practice.

Regarding the effectiveness of risk management, most respondents (48.86%) believe that it depends on the project team profile. The respondents justified that Scrum can bring better results in teams with great experience in risk management. However, 33.33% of respondents believe that risk management in Scrum is more effective, independently of team's profile, because the risks are treated more often during the framework ceremonies.

According to 38.10% of respondents, the integration of traditional practices of risk management to Scrum would be effective because the use of specific practices of Risk Management would improve this process in Scrum projects. On the other hand, 28.57% of respondents believe that the effectiveness of this integration would be low, because the Scrum approach differs from that presented by traditional

methods, which would leave the work heavy and slow rather than agile.

For 23.81% of the respondents believe that the effectiveness of this integration would depend on how it is performed. The adaptation of risk management is necessary to preserve principles presented by the Agile Manifesto [55]. Another justification of the respondents is that traditional risk management methodologies would bring greater efficiency to the Scrum depending on project risk, where higher risk projects would benefit from this integration.

V. CONCLUSIONS

This research identified that the two most common Risk Management practices in Scrum projects are related to aspects of communication and individual behavior. On the other hand, the two less common identified practices in Scrum refers to the implementation of formal processes and documentation and to the Scrum Master responsibilities. Furthermore, Scrum projects should not have formal planning and the Scrum Master should not focus on the implementation of formal processes.

The analyzed results indicate that the Risk Management performed in Scrum projects is different of its application proposal on traditional methodologies. The framework possess native resources that contribute to risks identification and follow-up frequently, but the effectiveness of Risk Management in Scrum projects would depend on the team's experience.

The other classic Risk Management processes, such as, planning, qualitative and quantitative analysis and risks response plans, must be incorporated and adapted to Scrum. It is important this adaptation to be performed preserving the benefits from this framework and The Agile Manifesto principles. Since it is an empiric framework, the effective Risk Management application on Scrum relies on the project team's maturity. This scenario is empowered by the absence of some or many Risk Management formal processes.

For the accomplishment of further researches, it is suggested to carry out grouping of the Risk Management practices according to artifacts, ceremonies and roles on Scrum. Can also be carried out the identification of the Scrum items which most contribute to Risk Management, performing the mapping and highlighting point of attention on dealing with project risks.

ACKNOWLEDGMENTS

The authors need to express their acknowledgments to three Brazilian research agencies: the CAPES Foundation, CNPq and FAPEMIG, and especially all interviewees and reviewers. We also thank the anonymous SEKE reviewers whose comments and suggestions brought us to an improved version of this paper.

REFERENCES

- [1] P.L Bannerman, "Risk and risk management in software projects: A reassessment" in *The Journal of Systems and Software*, vol. 81, n. 12, 2008, pp. 2118-2133.
- [2] R. Kaur and J. Sengupta, "Software Process Models and Analysis on Failure of Software Development Projects" in *International Journal of Scientific & Engineering Research*, vol. 2, n. 2, 2011, pp. 1-4.
- [3] Forrester Research, "Software and Services in Large Enterprises" in *Business Technographics*, Forrester Research, 2005.
- [4] D. West and T. Grant, "Agile Development: Mainstream Adoption Has Changed Agility" in *Forrester Research*, 2010.
- [5] J. Erickson, K. Lyytinen, and K. Siau, "Agile Modeling, Agile Software Development, and Extreme Programming: The State of Research" in *Journal of Database Management*, vol. 16, n. 4, 2005, pp. 88-100.
- [6] E.T. Alharbi and M.R.J. Qureshi, "Implementation of Risk Management with SCRUM to Achieve CMMI Requirements" in *I.J. Computer Network and Information Security*, vol. 11, 2014, pp. 20-25.
- [7] J. Garzás and M.C. Paulk, "A case study of software process improvement with CMMI-DEV and Scrum in Spanish companies" in *Journal of Software: Evolution and Process*, vol. 25, n. 12, 2013, pp. 1325-1333.
- [8] Z. Racheva, M. Daneva, "Reprioritizing the Requirements in Agile Software Development: towards a Conceptual Model from Clients' Perspective" in *The 21th International Conference on Software Engineering & Knowledge Engineering*, 2009, pp. 73-80.
- [9] R.N. Charette, "Why software fails" in *IEEE Spectrum*, vol. 42, n. 9, 2005, pp. 42-49.
- [10] A.A.M. Chowdhury and S. Arefeen, "Software Risk Management: Importance and Practices" in *International Journal of Computer and Information Technology (IJCIT)*, vol. 02, n. 01, 2011, pp. 49-54.
- [11] SEI - Software Engineering Institute. CMMI - Capability Maturity Model Integration. Version 1.3, Pittsburgh, PA, Carnegie Mellon University, USA, 2010.
- [12] B. de Wet and J.K. Visser, "An evaluation of software project risk management in South Africa" in *South African Journal of Industrial Engineering*, vol. 24, n. 1, 2013, pp. 14-28.
- [13] S.K. Khatri, K. Bahri, and P. Johri, "Best Practices for Managing Risk in Adaptive Agile Process" in *International Conference on Reliability, Infocom Technologies and Optimization (ICRITO)*, 2014, pp. 1-5.
- [14] A. Moran, "Agile Risk Management" in *SpringerBriefs in Computer Science*, 2014, pp. 33-60.
- [15] J. Nyfjord and M. Kajko-Mattsson, "Outlining a model integrating risk management and Agile software development" in *Proceedings of the 34th EUROMICRO Conference on Software Engineering and Advance Applications IEEE Computer Society*, 2008, pp. 476-483.
- [16] E. Odzaly, D. Greer, D. Stewart, *Lightweight Risk Management in Agile Projects in: The 26th International Conference on Software Engineering & Knowledge Engineering*, 2014, pp. 576-581
- [17] H. Hijazi, T. Khmour, A. Alarabeyyat, "A Review of Risk Management in Different Software Development Methodologies" in *International Journal of Computer Applications*, vol. 45, n. 7, 2012, pp. 8-12.
- [18] B.G. Tavares, "Risk Management Analysis in Software Projects that Use the Scrum Framework", *Masters Dissertation - Federal University of Itajubá, Itajubá*, 2015.
- [19] P.G. Smith and R. Pichler, "Agile Risks, Agile Rewards" in *Software Development Magazine*, vol. 13, 2005, pp. 50-53.
- [20] A.S.C. Marçal, B.C.C Freitas, F.S.F Soares, A.D. Belchior, "Mapping CMMI Project Management Process Areas to SCRUM Practices" in *Software Engineering Workshop*, 2007 pp. 13-22.
- [21] J.R. Menezes, C.M.G Gusmão, and H. Moura, "Defining Indicators for Risk Assessment in Software Development Projects" in *CLEI Electronic Journal*, vol. 16, n. 1, 2013.
- [22] C.R. Jakobsen and K.A. Johnson, *Mature Agile with a Twist of CMMI in Agile '08 Conference*, 2008.
- [23] V. Szalvay, "Glossary of Scrum Terms", Available at: <https://www.scrumalliance.org/community/articles/2007/march/glossary-of-scrum-terms#1126>. Accessed on: February 09th 2016.
- [24] M. Tomanek and J. Juricek, "Project Risk Management Model Based on PRINCE2 and Scrum Frameworks" in *The International Journal of Software Engineering & Applications (IJSEA)*, vol. 6, n. 1, 2015, pp. 81-88.
- [25] K. Schwaber and J. Sutherland Jeff, "The Scrum Guide" in *Scrum.org*, 2013.
- [26] S.P. Ravi, B. Reddaiah, L.S. Movva, and R. Kilaparthi, "A Critical review and empirical study on success of risk management activity with respect to Scrum" in *Engineering Science and Technology: An International Journal (ESTIJ)*, vol. 2, n. 3, 2012, pp. 467-473.
- [27] European Comission. *User guide to the SME definition. Internal Market, Industry, Entrepreneurship and SMEs*, Maio, 2015.
- [28] A.J. Shenhar and D. Dvir, "Reinventing Project Management: the diamond approach to successful growth and innovation" in *Harvard Business School Press*, 2007.
- [29] M. Silva and C. Jeronimo. "From Taylor to Tailoring - In Pursuit of the Organizational Fit" in *Second International Scientific Conference on Project Management in the Baltic Countries*, 2013.
- [30] B.J. Sausser, R. Reilly, and A.J. Shenhar, "Why projects fail? How contingency theory can provide new insights - A comparative analysis of

- NASA's Mars Climate Orbiter loss" in *International Journal of Project Management*, vol. 27, n. 7, 2009, pp. 665-679.
- [31] D. Dvir, A. Sadeh, and A. Malach-pines, "The fit between entrepreneurs' personalities and the profile of the ventures they manage and business success: An exploratory study" in *The Journal of High Technology Management Research*, vol. 21, n. 1, 2010, pp. 43-51.
- [32] M. Frank, A. Sadeh, and S. Ashkenasi, "The Relationship Among Systems Engineers' Capacity for Engineering Systems Thinking, Project Types, and Project Success" in *Project Management Journal*, vol. 42, n. 5, 2011, pp. 31-41.
- [33] B. Vanathi, T. Saravanan, and M. Nagarajan, "Growth of Literature in Chemistry Research Output in Tamil Nadu Universities: A Scientometric Study (1989 -2015)" in *Journal of Advances in Library and Information Science*, vol. 43, n. 3, 2015, pp. 187-190.
- [34] S. Mukherjee, B. Uzzi, B. Jones, and M. Stringer, "A New Method for Identifying Recombinations of Existing Knowledge Associated with High-Impact Innovation" in *Journal of Product Innovation Management*, vol. 33, n. 2, 2016, pp. 224-236.
- [35] P. Muñoz, M. Ontiveros, S. De La Barquera, and A. L. Packer, "Action Lines for the Years 2014-2016 with the Objective of Increasing the Visibility of the SciELO Network Journals and Collections" in *Reunião sobre linhas de ação para o desenvolvimento dos periódicos indexados no SciELO. Santiago de Chile, 2013*, pp. 25-27.
- [36] J. M. Swales and C. Leeder, "A reception study of the articles published in English for Specific Purposes from 1990-1999" in *English for Specific Purposes*, vol. 31, 2012, pp. 137-146.
- [37] J. Cusker, "Elsevier Compendex and Google Scholar: A Quantitative Comparison of Two Resources for Engineering Research and an Update to Prior Comparisons" in *The Journal of Academic Librarianship*, vol. 39, n. 3, 2013, pp. 241-243.
- [38] J. F. Hair, B. Babin, A.H. Money, and P. Samouel, "Fundamentos de métodos de pesquisa em administração", Porto Alegre, Bookman, 2005.
- [39] Y. Kwak and J. Stoddard, "Project risk management: lessons learned from software development environment" in *Technovation*, vol. 24, n. 11, 2004, pp. 915-920.
- [40] A. Bryman and E. Bell, "Business research methods", 2^a ed., New York: Oxford University Press, 2007.
- [41] E.G. Carmines and R.A. Zeller, "Reability and Validity Assessment", Londres, Sage, 1979.
- [42] D.R. Heise and G.W. Bohrnstedt, "Validity, Invalidity and Reliability", in EF Borgatta, G Bohrnstedt (Eds.). *Sociological methodology*, 1970, pp. 104-129.
- [43] C. Fan, Y. Yu, "BBN-based software project risk management" in *The Journal of Systems and Software*, n. 73, 2004, pp. 193-203.
- [44] M.O. Barros, C.M.L Werner, and G.H. Travassos, "Supporting risks in software project management" in *The Journal of Systems and Software*, vol. 70, n. 1-2, 2004, pp. 21-35.
- [45] H. Barki, S. Rivard, and J. Talbot, "An Integrative Contingency Model of Software Project Risk Management" in *Journal of Management Information Systems*, vol. 17, n. 4, 2001, pp. 37-69.
- [46] T. Dingsøy, G.K. Hanssen, and T. Dybå, "Developing Software with Scrum in a Small Cross-Organizational Project" in *Proceedings of the 13th European conference on Software Process Improvement, Joensuu, Finland, 2006*, pp. 11-13.
- [47] N.B. Moe, T. Dingsøy, T. Dybå, "A teamwork model for understanding an agile team: A case study of a Scrum project" in *Information and Software Technology*, n. 52, 2010, pp. 480-491.
- [48] T. Dyba and T. Dingsøy, "Empirical studies of agile software development: A systematic review" in *Information and Software Technology*, vol. 50, n. 9-10, 2008, pp. 833-859.
- [49] C. Felker, R. Slamova, J. Davis, "Integrating UX with Scrum in an Undergraduate Software Development Project" in *SIGCSE '12 Proceedings of the 43rd ACM technical symposium on Computer Science Education, 2012*, pp. 301-306.
- [50] J. Ahola, C. Frühwirth, M. Helenius, L. Kutvonen, J. Myllylahti, T. Nyberg, A. Pietikäinen, P. Pietikäinen, J. Röning, S. Ruohomaa, C. Särs, T. Siiskonen, A. Vähä-Sipilä and V. Ylimannela, "Handbook of The Secure Agile Software Development Life Cycle", University of Oulu, 2014.
- [51] R. Hoda, P. Kruchten, J. Noble, and S. Marshall. "Agility in Context" in *Proceedings of the ACM International Conference on Object Oriented Programming Systems, Languages, and Applications (OOPSLA), 2010*, pp. 74-88.
- [52] O. Ktata and G. Lévesque, "Designing and Implementing a Measurement Program for Scrum Teams: What do agile developers really need and want?" in *3rd Conference on Computer Science and Software Engineering (C3S2E), Montreal, QC, Canada. Association for Computing Machinery, 2010*, pp. 101-107.
- [53] M. Lárusdóttira, Á. Cajanderb, and J. Gulliksen, "Informal feedback rather than performance measurements – user-centred evaluation in Scrum projects" in *Behaviour & Information Technology*, vol. 33, n. 11, 2014, pp. 1118-1135.
- [54] M.R.J Qureshi, and A. Albarqi, "Proposal of New PRORISK Model for GSD Projects" in *International Journal of Information Technology and Computer Science*, vol. 6, 2015, pp. 38-44.
- [55] The Agile Manifesto. Principles behind the Agile Manifesto. Available at: <http://www.agilemanifesto.org/principles.html>. Accessed on: February 20th 2016.

DmS-Modeler: A Tool for Modeling Decision-making Systems for Self-adaptive Software Domain

Frank José Affonso, Gustavo Leite
Dept. of Statistics, Applied Mathematics and Computation
Univ Estadual Paulista - UNESP
Rio Claro, SP, Brazil
frank@rc.unesp.br, gustavoleite.ti@gmail.com

Elisa Yumi Nakagawa
Dept. of Computer Systems
University of São Paulo - USP
São Carlos, SP, Brazil
elisa@icmc.usp.br

Abstract—The ability to modify its own structure and/or behavior at runtime is a native feature in the development of Self-adaptive Software (SaS). In previous work, a Reference Architecture for SaS (RA4SaS), an automated process for adaptation, and a framework for decision-making were developed to assist the development of SaS. Although such initiatives have collaborated with evolution of SaS, the design of the Decision-making Systems (DmS), element of first class for SaS, is manually conducted. Therefore, this paper presents a tool called *DmS-Modeler*, which aims to assist the development of DmS for SaS, providing facilities for modeling, calibration of such system, and automatic generation of infrastructure (i.e., source code and databases). Aiming to present the applicability of our tool, a case study was conducted and the results enable us to have good perspectives of contribution to the SaS area and other domains of software systems.

Keywords—Self-adaptive software; Reference Architecture; Tool; Decision-making System.

I. INTRODUCTION

The complexity of software systems and their computational environments has increased in the last years. In general, our society is becoming increasingly dependent of such systems, which must be able to work in 24/7 mode (i.e., 24 hours per day, seven days per week). Moreover, most of them must be prepared to operate in adverse conditions, maintaining their integrity of execution. Therefore, features as robustness, reliability, scalability, and flexibility have been increasingly required by these systems in the presence of changes (e.g., new needs of their users and/or modifications in the execution environment). These features and examples have been associated with the area of Self-adaptive Software (SaS), which are systems capable of self-configuration, self-adaptation and self-healing, self-monitoring and self-tuning, and so on. In short, they enable the incorporation of new features or functionalities at runtime without interruption in their execution [1], [2].

Software adaptation, when manually performed, becomes an onerous (e.g., regarding time, effort, and money) and error-prone activity (e.g., involuntary injection of uncertainties by the developers) [3], [4]. To overcome these adversities, automated approaches have been adopted as a feasible alternative

to maximize the speed of SaS implementation and, at the same time, minimize the developers' involvement [4], [5].

In parallel, Reference Architectures (RA) refer to a special class of software architecture that have become an important element to systematically reuse architectural knowledge [6], [7]. Thus, in previous work [3], [8] we have proposed a Reference Architecture for SaS (RA4SaS)[8] – an architecture that provides a set of guidelines for the SaS development and an automated process for self-adaptation of the software entities at runtime without human intervention. From this point onwards, SaS may be also referred to as software entities or simply entities. Moreover, as part of this RA, a framework for decision-making was developed [9], whose purpose is to identify anomalies and propose one or more solutions to solve them. The RA and this framework have improved the SaS development; however, the design of Decision-making Systems (DmS), i.e., instances of such framework, was still manually conducted. According to Whitehead [5], Nakagawa et al. [6] and Gray [10], software engineering tools, among several purposes, can significantly reduce the time and development cost in a software project. In this sense, the main contribution of this paper is to present a tool called *DmS-Modeler*, which aims to support the DmS development for SaS. As result, to the best of our knowledge, there is no other complete solution as RA4SaS that enables the SaS development in both design and infrastructure.

The paper is organized as follows: Section II presents the background and related work; Section III provides a description of RA4SaS and the automated process for SaS adaptation; Section IV shows the *DmS-Modeler* tool and its approach of development; Section V presents a case study to show the applicability of our tool; and Section VI summarizes the contributions and presents perspectives for further research.

II. BACKGROUND AND RELATED WORK

This section presents the background (i.e., concepts and definitions on self- \star systems and RA) and related work on our paper.

Self- \star Systems. SaS has specific features in comparison to traditional one because this type of software system constantly deals with structural and/or behavioral changes at runtime. Some of them deal with management of complexity, robustness in handling unexpected conditions (e.g., failure), changing priorities and policies governing the goals, and context conditions (e.g., execution environment). The SaS development has boosted self- \star properties in general-purpose software systems, such as self-managing, self-configuring, self-organizing, self-protecting, self-healing, and so on. These properties allow systems to automatically react against users' needs or to respond as soon as these systems meet execution environment changes [1], [11]. According to Silva and De Lemos [12], there is a set of goals to be achieved so that structural and behavioral modifications are performed in the SaS without impacting its execution states. For these authors, an adaptation plan is a feasible solution to define which procedures will be adopted so that such changes are implemented.

Reference Architecture. According to Nakagawa et al. [7], RAs refer to a special type of software architecture that have become an important element to systematically reuse architectural knowledge. The main purpose of these architectures is to facilitate and guide [7]: (i) the design of concrete architectures for new systems; (ii) the extensions of systems of neighbor domains of a RA; (iii) the evolution of systems that were derived from the RA; and (iv) the improvement in the standardization and interoperability of different systems. Considering their relevance for the software development, different domains have proposed RAs. For instance, service-oriented architectures such as IBM's foundation architecture [13] and architectures for software engineering environments [6] are some of RAs found in the literature. Other areas/domains have also proposed RAs, including the self- \star software (e.g., RA4SaS [8]).

As related work, Schneider et al. [14] presented a survey on frameworks for self-healing systems. According to this study, these systems can combine machine learning techniques and control loops to reduce human intervention, since such systems can autonomously detect and recover themselves from faulty states. The authors proposed a classification of self-healing frameworks in three categories: (i) learning methodology (supervised, semi-supervised, and unsupervised); (ii) management style (bottom-up and top-down); and (iii) computing environment (n-tier traditional, cloud, virtualized, and grid/p2p). In Psailer and Dustdar [15], a survey on self-healing systems was conducted. This study showed that the number of approaches for the research on self-healing has been very active. Moreover, a selection of current and past self-healing approaches were addressed, as well as explanations for the origins, principles, and theories of self-healing for such approaches. These two studies provided the theoretical basis for the design of our framework [9].

Qun et al. [16] reported that architecture-based self-healing approaches were used in the architectural models as basis for system adaptation. Such approaches were based on architectural reflection so that their software architectures are

observable and controllable. Cheng et al. [17] proposed a software architecture-based adaptation for grid computing. Technically, the study designed a framework based on a software architectural model, which allows the analysis of the adaptation needs in an application, enabling repairs to be written in the context of the architectural model and propagated to the running system. Zadeh and Seyyedi [18] suggested an architecture based on failure-prediction in architectures based on web services. The main goal of this study is to repair the execution process after detection of a failure. In a similar context, Psailer et. al [19] developed a self-healing approach that enables recovering mechanisms to avoid degraded or stalled systems. As result a framework called VieCure was designed to support self-healing principles in mixed service-oriented systems. In this context, one can highlight that the literature has revealed important initiatives for the context of this paper.

III. REFERENCE ARCHITECTURE FOR SAS

According to Affonso and Nakagawa [8], the RA4SaS (Figure 1) is composed of four external modules and a core of adaptation. This RA works with a controlled adaptation approach, i.e., the software engineer must insert annotations in each software entity so that the automatic mechanisms in the environment execution can identify the adaptation level of each entity. These levels contain parameters that determine where the new changes may be applied. Thus, when an entity is developed, an automatic mechanism performs a scan process, to inspect if such annotations were correctly inserted. After a validation process, these entities are inserted in the execution environment (i.e., entity repositories) so that they may be invoked in future adaptations. Next, a brief description of this architecture is presented.

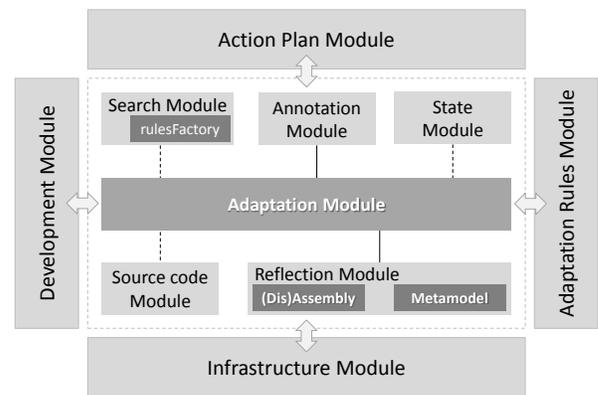


Fig. 1. General View of the RA4SaS [8]

The **Development Module** provides a set of guidelines for requirement analysis, design, implementation, and adaptation of the software entities at runtime. The **Action Plan Module** aims at assisting in the adaptation of such entities, providing controls for: (i) dynamic behavior, (ii) individual reasons, and (iii) execution state in relation to the environment. Thereby, a

framework based on learning techniques [20] and the MAPE-K loop [21], [22] was developed in previous work as part of this module to support the decision-making [9]. Section IV presents details on this framework as well as the design and implementation of the *DmS-Modeler* tool. The **Adaptation Rules Module** provides a set of rules (i.e., metrics) for adaptation of the software entities. The **Infrastructure Module** provides support for the adaptation of software entities at runtime, i.e., a set of mechanisms for the dynamic compiling and dynamic loading of such entities. Finally, the **Core of Adaptation** represents an automated process composed of a logic sequence of nine well-defined steps so that the adaptation of the software entities is conducted with no human intervention [8].

IV. *DmS-Modeler* TOOL

As presented in Section III, the RA4SaS has an automated process to accommodate the structural or behavioral changes in a software entity. According to authors of this RA [8], the modification of a software entity is a complex activity, since an action plan must be established so that the software entities are adapted. In short, this plan must be elaborated based on new requirements (e.g., structural and/or behavioral) and adaptation level of each software entity. Based on this scenario, a framework for decision-making was developed by Affonso et al. [9] to support the generation of such plan identifying anomalies and proposing one or more solutions to solve them. Concerning the design, MAPE-K loop [22], learning techniques [23], [24], and an external approach [4] of adaptation were the resources used in this framework.

The MAPE-K loop is a reference model (i.e., mature solution) that has been adopted to provide an autonomous behavior in SaS. In short, the *Monitor* process receives data collected from sensors and convert such data in “symptoms”. The *Analyze* process aims to correlate the collected data and to model complex situations so that the autonomous systems can learn from the environment and predict future situations. The *Plan* process is responsible to create a plan for adaptation, i.e., what will be adapted and how to apply the changes in a software entity. The *Execute* process must provide the mechanisms that can execute the action plan (i.e., proposed solutions). *Sensors* and *Effectors* are components to generate a collection of data reflecting the system state to rely on in vivo mechanisms or autonomous subsystems to apply changes [4], [21], [22].

Concerning the learning technique, an incremental classifier and association rules were used in the design of the classification and recommendation modules. The incremental classifier analyzes the training data and produces an inferred function, which can be used for mapping new instances [23], [24]. The association rule [23] aims to detect more significant statistically correlations, via support and confidence, among the occurred changes in order to operate the recommendation of solutions for such changes [24]. It is worth noting that there is no interest in a specific attribute (i.e., specific solution), since a set of changes may present a set of solutions.

Finally, the external adaptation approach enables the organization of a SaS in two layers: (i) adaptation engine, which contains the logical for adaptation; and (ii) adaptable software, which represents entities that can be adapted. Therefore, our framework [9] acts as a non-intrusive supervision modality, i.e., a supervisor system (engine) can be coupled to a software entity (adaptable software) to monitor its internal state of operation or the execution environment in which it is inserted.

Although our framework has been designed to assist the process of decision-making and propitiate its reuse in different systems (e.g., SaS and other software domains), the development of a DmS is manually conducted and, hence, it can be considered onerous and error-prone activity. Therefore, the design of a tool that meets the following requirements is a feasible alternative: (i) modeling of the problem and definition of the main points of monitoring for a software system (SaS); (ii) automatic generation of the classification and recommendation modules; (iii) insertion of initial knowledge into such modules; and (iv) data calibration of the DmS. According to these requirements, a tool called *DmS-Modeler* was developed and a set of guidelines elaborated to support the development of DmS for SaS, as illustrated in Figure 2. In short, the development of this system type is composed of three phases: design, execution, and instruction mapping. The first is organized in two modeling steps: (a) *classification module*, which main purpose is to present a classification for a data set collected via sensors from execution environment; and (b) *recommendation module*, which aims to present a solution set ranked by statistical measures for a problem reported by the classification module. The second represents the complete instantiation of the DmS for the calibration process into its execution environment. The third consists of mapping knowledge (i.e., recommendation module) to instructions in source code. Next, a description of each step is reported.

During the design of the classification module, the stakeholders (i.e., software engineer and domain specialist) must define the “main points” of monitoring for a software entity. These points represent the attribute number of an instance and values that can be assigned to them. In this sense, the *DmS-Modeler* tool has a wizard (i.e., a visual front-end) that enables the creation of attributes and their values, which may be numerical or nominal. In order to make compatible the values collected from the execution environment in relation to ones required by the algorithms used in the implementation of both modules, a discretization process of values was implemented in our tool. After the definition of attributes and values, the stakeholders must save the specification of the DmS in a “.arff” file, which is represented by the “Specification Repository” component. This file will be used by our tool to generate the database (“Problems” component) of each system. Based on this specification and a set of labeled initial data, an incremental classifier is generated in the “Classifier” component. Next, new data can be collected from execution environment and sent to this classifier for identification of anomalies. The data is stored in the database as collected and classified after the validation by the test module [9].

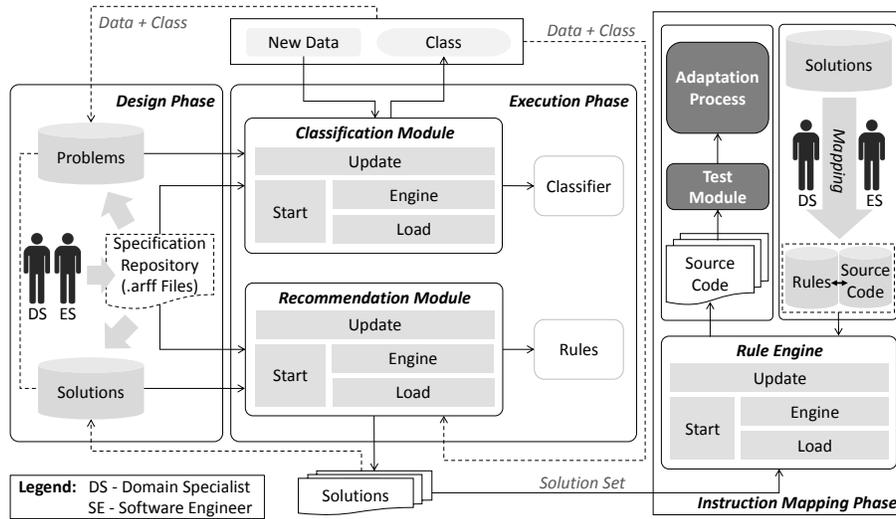


Fig. 2. Development phases of the *DmS-Modeler* tool

The design of the recommendation module must be conducted in a similar way to previous one [9]. The stakeholders must conduct the specification of the solutions and stored them in the “Specification Repository” component as a “.arff” file, which will be used by the *DmS-Modeler* tool to generate the database (“Solutions” component). The insertion of labeled initial data is different from the previous module, since the stakeholders must provide one or more solution for each instance of problem. From these databases (i.e., Problems and Solutions) and specifications are generated a rule set (“Rules” component), which intends to map the problems and solutions. The design of the rule engine is conducted in parallel to this module, since the main purpose of this engine is to provide a solution in source code for each instance of problem. Thus, each one must be labeled by a class to receive one or more solution in source code (“Source Code” repository), generating a rule in the “Rules” repository. Before it becomes an effective solution, a recommendation must be tested (“Test Module”) in order to ensure that no “collateral effects” will be propagated to the software system. For space and scope reasons, the design of this module will not be detailed in paper.

Regarding the internal components of each module, a brief description is addressed [9]: (i) **Start**: aims to initialize each module by means of the “Engine” component. As result, an incremental classifier (“Classifier” component), rule set (“Rules” component), and a rule engine are generated based on the specifications and initial knowledge provided by the specialist; (ii) **Load**: attempts to load data stored in each database, which is organized in two types: (a) specification, i.e., initial knowledge provided by the specialist; and (b) acquired knowledge, i.e., data obtained during the execution cycle; (iii) **Engine**: represents an abstraction for the algorithms of classification and recommendation, which were developed by third parties; and (iv) **Update**: updates the database after new data has classified and validated. Such update is performed when a notification from the test module is received. Finally, it is

noteworthy that our tool has a flexible engine, since it enables that other algorithms can be coupled to it without additional implementation in its modules.

V. CASE STUDY

In this section we present a case study to evaluate the applicability of the *DmS-Modeler* tool. The main purpose of this study is to demonstrate the real value of our tool for the stakeholders during the development of a DmS. In previous work, we have applied our framework for the monitoring and eventual corrections of flight plan for Unmanned Aerial Vehicles (UAVs) [9], since this system type requires unforeseen context changes. Therefore, we will approach the same system in this paper; however emphasizing the use of the *DmS-Modeler* tool. Next, a brief description of our subject application and the empirical strategies adopted for conducting this case study is presented.

Subject Application. For our empirical analysis we have selected an application addressed to the management of an UAV in a simulated environment. In short, this application is organized in three layers: (i) *UAV*, which is composed of a set of UAVs; (ii) *Communication*, which contains the servers for communication between UAVs and clients; and (iii) *Client*, which represents the controllers of the UAVs in different operating systems. The UAVs used in the scope of this empirical study are equipped with five sensors: (i) altitude, (ii) battery level, (iii) distance, (iv) speed, and (v) temperature. These sensors provide numerical information that must be discretized, since both algorithms of our tool require data in the nominal form [20]. Operationally, this application enables us to collect data from the environment via sensors, and transferring it for classification. Thus, when a problem is detected, a set of useful solutions is presented for correcting the flight plan. In extreme cases, the system may exhibit a recommendation to abort the operation. This last case is recommended when the UAV integrity may be compromised.

Then, the UAV location is provided for our system, enabling the vehicle to be rescued. Modifications are made in the flight plan when the collected data tell us that something unplanned is changing in the environment. Thus, even if no decision is taken, the mission of the UAV may be compromised.

Empirical research strategy. Figure 3 illustrates the systematization for modeling of a DmS, which is organized in two steps: (i) modeling of the problem; and (ii) calibration of the modules. Initially, the stakeholders must identify the monitoring points for such system (Step 1). Next, for each point, a verification is conducted to obtain its output (i.e., *Is it a numerical or nominal value?*). The main difference between the modeling of numerical and nominal values consists of definition of a label (i.e., nominal value) for an interval of numerical values, as illustrated in the table fragment “battery level sensor” (Step 3.1). For space reasons, only the discretization of this sensor will be presented. The first column shows the range to classify the battery level (second column) on a scale of six to seven percentage points (i.e., A with six points and B and C with seven points). The third column presents a classification in a scale of 20 points in relation to first column. We also defined that a classification has three levels, i.e., the A level is the best state of a classification, the B level can be considered as a region of stability, and the C level represents a transition stage. Next, we combine the first letter of each classification with respective battery charge levels to create a nominal category, as shown in column 4. After discretization, each sensor must be transformed in an attribute and its label in value for it. Step 3.2 represents the insertion of monitoring points (i.e., attributes) by the stakeholders into our tool. Step 4 represents the generation of a complete infrastructure for the classification module (i.e., database containing all the attributes and source code for this module). Step 5 represents the mapping of solutions for each type of problem. In this wizard, the stakeholders can select the project developed in the previous step so that the recommendation module is generated (Step 6). Next, an initial knowledge must be provided to the databases of the classification and recommendation modules (Step 7). Our tool provides a table for insertion of knowledge for the classification module and a list for mapping of solutions. Each line of this table represents a system state in the execution environment. To overcome possible adversities during its execution cycle, one or more solutions can be created and associated to each state to maintain a system in execution. Our tool also enables the validation of the inserted knowledge by means of a wizard (Step 8), which aims to ensure that a problem has one or more solution. After the insertion of knowledge, the calibration process can be conducted to evaluate the precision of a classification of problems and the recommendation of solutions [9], [15], [19], [20].

VI. CONCLUSIONS AND FUTURE WORK

This paper presented *DmS-Modeler* tool that intends to support the SaS development. This tool incorporates a framework for decision-making [9] and, hence, the benefits of our tool can also be extended for other communities of software

development. As reported in this paper, an approach was proposed to systematize the development of a DmS using our tool. Based on this scenario, the main contributions of this paper are:

- For the SaS area by providing a means that facilitates the development of DmS for SaS;
- For the use of our framework [9], providing a wizard for modeling of the monitoring point. For instance automatic generation of the classification and recommendation modules, insertion of initial knowledge, and calibration of the DmS are other functionalities provided by our tool that can be executed by means of a wizard. Moreover, the automation of the activities for the software development tends to minimize human involvement and involuntary generation of uncertainties, which are considered negative factors for the software development, especially for the SaS [5], [6], [8], [10], [9]; and
- For the RA area, since we have proposed the first RA based on reflection [8] and the implementation of this tool aims to optimize our initiative for the development of such systems.

As future work, three goals are intended: (i) conduction of more case studies intending to completely evaluate our tool, including different software domains; (ii) implementation of a new wizard for mapping of solutions to instructions (i.e., source code), which would complete the development cycle presented in Figure 2; and (iii) use of this tool in the industry, since it is intended to evaluate its behavior when it is applied in larger real environment of development and execution. Therefore, it is expected that a positive scenario of research, intending to have this tool become an effective contribution to the software development community.

ACKNOWLEDGMENT

This research is supported by PROPE/UNESP and Brazilian funding agencies (FAPESP, CNPq and CAPES).

REFERENCES

- [1] J. Kramer and J. Magee, “Self-managed systems: an architectural challenge,” in *FOSE’ 07*, 2007, pp. 259–268.
- [2] P. Maes, “Concepts and experiments in computational reflection,” *SIGPLAN Notice*, vol. 22, no. 12, pp. 147–155, December 1987.
- [3] F. J. Affonso and E. L. L. Rodrigues, “A proposal of reference architecture for the reconfigurable software development,” in *SEKE’ 12*, 2012, pp. 668–671.
- [4] M. Salehie and L. Tahvildari, “Self-adaptive software: Landscape and research challenges,” *ACM Transactions on Autonomous and Adaptive Systems*, vol. 4, no. 2, pp. 1–42, 2009.
- [5] J. Whitehead, “Collaboration in software engineering: A roadmap,” in *Future of Software Engineering*, 2007, pp. 214–225.
- [6] E. Y. Nakagawa, F. C. Ferrari, M. M. F. Sasaki, and J. C. Maldonado, “An aspect-oriented reference architecture for software engineering environments,” *Journal of Systems and Software*, vol. 84, no. 10, pp. 1670–1684, 2011.
- [7] E. Y. Nakagawa, F. Oquendo, and M. Becker, “RAModel: A reference model of reference architectures,” in *WICSA/ECSA’ 12*, 2012, pp. 297–301.
- [8] F. J. Affonso and E. Y. Nakagawa, “A reference architecture based on reflection for self-adaptive software,” in *SBCARS’ 13*, 2013, pp. 129–138.
- [9] F. J. Affonso, G. Leite, R. A. P. Oliveira, and E. Y. Nakagawa, “A framework based on learning techniques for decision-making in self-adaptive software,” in *SEKE’ 15*, 2015, pp. 1–6.

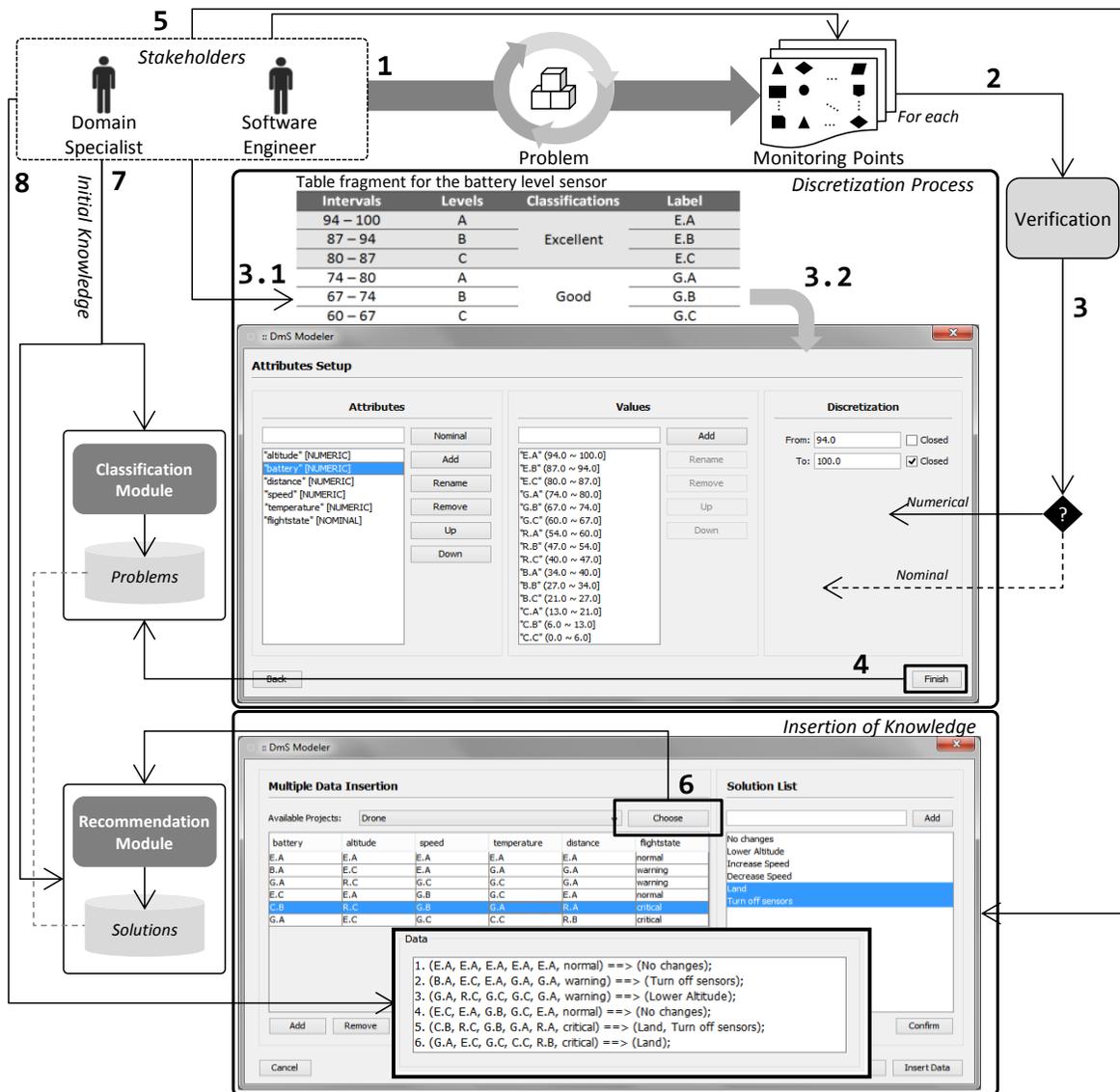


Fig. 3. Case study

- [10] J. Gray, "Software engineering tools," in *HICSS' 00*, January 2000, pp. 3300–3301.
- [11] D. Weyns, S. Malek, and J. Andersson, "Forms: a formal reference model for self-adaptation," in *ICAC' 10*, 2010, pp. 205–214.
- [12] C. E. Silva and R. Lemos, "A framework for automatic generation of processes for self-adaptive software systems," *Informatica Journal*, vol. 35, no. 1, pp. 3–13, 2011.
- [13] R. High, S. Kinder, and S. Graham, "Ibm's soa foundation - an architectural introduction and overview," 2005, available in: <http://signallake.com/innovation/soaNov05.pdf> (Access on March 1, 2016).
- [14] C. Schneider, A. Barker, and S. Dobson, "A survey of self-healing systems frameworks," *Software: Practice and Experience*, pp. n/a–n/a, 2014.
- [15] H. Psaiar and S. Dustdar, "A survey on self-healing systems: Approaches and systems," *Computing*, vol. 91, no. 1, pp. 43–73, January 2011.
- [16] Y. Qun, Y. Xian-chun, and X. Man-wu, "A framework for dynamic software architecture-based self-healing," in *SMC' 05*, vol. 3, October 2005, pp. 2968–2972 Vol. 3.
- [17] S.-W. Cheng, D. Garlan, B. Schmerl, P. Steenkiste, and N. Hu, "Software architecture-based adaptation for grid computing," in *HPDC' 02*, 2002, pp. 389–398.
- [18] M. H. Zadeh and M. A. Seyyedi, "A self-healing architecture for web services based on failure prediction and a multi agent system," in *ICADIWT' 11*, August 2011, pp. 48–52.
- [19] H. Psaiar, F. Skopik, D. Schall, and S. Dustdar, "Behavior monitoring in self-healing service-oriented systems," in *COMPSAC' 10*, July 2010, pp. 357–366.
- [20] P.-N. Tan, M. Steinbach, and V. Kumar, *Introduction to Data Mining, (First Edition)*. Boston, MA, USA: Addison-Wesley Longman Publishing, 2005.
- [21] S. Dobson, S. Denazis, A. Fernández, D. Gañti, E. Gelenbe, F. Massacci, P. Nixon, F. Saffre, N. Schmidt, and F. Zambonelli, "A survey of autonomic communications," *ACM Transactions on Autonomous and Adaptive Systems*, vol. 1, no. 2, pp. 223–259, December 2006.
- [22] IBM, "An architectural blueprint for autonomic computing," [Online], *World Wide Web*, 2005, available in: <http://www-03.ibm.com/autonomic/pdfs/AC%20Blueprint%20White%20Paper%20V7.pdf> (Access on March 1, 2016).
- [23] R. Agrawal, T. Imieliński, and A. Swami, "Mining association rules between sets of items in large databases," in *ACM SIGMOD' 93*, 1993, pp. 207–216.
- [24] C. Tew, C. Giraud-Carrier, K. Tanner, and S. Burton, "Behavior-based clustering and analysis of interestingness measures for association rule mining," *Data Mining and Knowledge Discovery*, vol. 28, no. 4, pp. 1004–1045, 2014.

Modeling and Analyzing Security Patterns Using High Level Petri Nets

Xudong He
School of Comp. and Inf. Sciences
Florida International University
Miami, FL 33199, USA

Yujian Fu
Department of EECS
Alabama A & M University
Normal, AL 35762, USA

Abstract – Security has become an essential and critical non-functional requirement of modern software systems, especially cyber physical systems. Security patterns aim at capturing security expertise in the worked solutions to recurring security design problems. This paper presents an approach to formally model and analyze six security patterns to detect potential incompleteness, inconsistency, and ambiguity in the textual descriptions; and to prevent their incorrect implementation. These patterns are modeled using high level Petri nets in our tool environment PIPE+. Simulation is used to analyze various security relevant properties. The validated formal models of individual security patterns serve as the building blocks for system design involving the composition of multiple security patterns.

Keywords – security patterns, formal modeling, high level Petri nets, validation, simulation

I. Introduction

Security has become an essential and critical non-functional requirement of modern software systems, especially cyber physical systems. However software developers often lack a deep understanding of system security issues and their proper solutions. Security patterns, evolved from general software design patterns, aim at capturing security expertise in the worked solutions to recurring security design problems. In the past decade, many security patterns and their description methods were proposed [1]. A comprehensive repository containing 26 (13 structural and 13 procedural) security patterns with 3 mini patterns was compiled in [2]. This repository collects some best known security patterns and provides an excellent starting point for developing secure software systems. The structural patterns contain descriptions of the structure and interactions of security assurance mechanisms, while the procedural patterns provide general guidelines for developing secure systems. Thus our focus is on structural patterns. Each structural pattern has a textual description with some graphical illustration to aid understanding. However none of the structural security patterns is formally modeled and analyzed since the majority of these structural patterns only contain generic descriptions

of general guidelines and thus are not suitable for precise definitions. However some pattern descriptions contain enough details for formal modeling. This paper presents an approach to formally model and analyze six of the above security patterns to detect potential incompleteness, inconsistency, and ambiguity in the text descriptions; and as a result to prevent their incorrect implementation. The validated formal models of individual security patterns serve as the building blocks for system design involving the composition of multiple security patterns.

II. Related Work

Despite many proposed security patterns and their descriptions in the past decade, very few work addressed the formal modeling and analysis of security patterns.

In [3], software design pattern template was adapted to describe 8 security patterns (1 creational, 2 behavioral, and 5 structural). The security patterns were modeled using UML diagrams, which are translated into Promela programs using tool Hydra [4]. The security constraints defined in linear time temporal logic formulas are checked using SPIN. A process of using the tool chain was presented and demonstrated through a simple example on check point pattern. While using UML diagrams can be helpful for implementation, they lack precise semantics and thus the analysis results based UML model translations may not be convincing. Furthermore, no detailed model checking result was given. In [5], a formal modeling approach for composing security patterns for web-based applications was proposed. This approach uses UML to model the security patterns and their composition and then transforms the UML model to Alloy (a formal specification language based on first order logic) formal specification for security property analysis. A case study of an on-line banking system was given, which involves five software security patterns: single sign on, check point, authenticator, policy, and secure proxy. No details of transforming a UML model to Alloy specification were given except some generic mappings. In [6], an approach of translating UML sequence diagrams to process algebra CCS was given. Two case studies were given, one involving the composition of secure pipe, authentication enforcer, and observer patterns and the other involving web security

pattern with third party brokered authentication. The behaviors of these compositions were first given in sequence diagrams, which were manually translated into CCS expressions. These CCS expressions serve as the behavior models. Various properties are defined using GCTL, which are model checked against CCS expressions using model checker CWB-NC. Unfortunately, the CCS expressions only capture the message names and control flows without considering the data processing and data flow; thus do not accurately reflect the complexity of real applications.

The above works only focus on the composition of security patterns while ignore the internal processing of individual security patterns. Our work focus on creating formal models from the textual descriptions of the internal processing of individual security patterns, and can be easily adapted to generate formal models from sequence diagrams representing security pattern composition.

III. Modeling Security Patterns

A. High Level Petri Nets

A high level Petri net [7] has a net structure $N = (P, T, F)$, which is a finite bipartite graph consisting of two kinds of nodes $P \cup T$, and the set of directed edges $F \subseteq P \times T \cup T \times P$. P is called the set of places, which are visually represented by circles; and T is called the set of transitions, which are visually represented by bars or boxes. The net structure (P, T, F) defines the *syntax* of a Petri net and models the control structure of a system. To define the static semantics, we need several semantic domains. First, we need a semantic domain of *Types*, which defines what are allowable tokens in each place. Elements of *Types* can be simple or composite, including Cartesian products and power set constructions. Second, we need a semantic domain *Labels*, which define permissible token flows. A label may contain variables to be instantiated under dynamic semantics. Third, we need a semantic domain of *Formulas*, which define the pre-conditions and post-conditions of transitions. A formula may contain variables to be instantiated under dynamic semantics. Finally, we need a semantic domain *Tokens* to define all valid tokens. The above semantic domains are defined in terms of an algebraic specification $Spec = (S, Op, Eq)$ with a family of sorts (types) S , the corresponding operations Op , and a set of equations Eq defining the meaning of the operations. Based on the above semantic domains, we can define the following semantic mappings:

- (1) $\varphi: P \rightarrow Types$ associates each place p in P with a type in *Types*. In a HLPN net, places are often called predicates to highlight their roles as in predicate logic.
- (2) $L: F \rightarrow Labels$ is a sort-respecting labeling of arcs.

- (3) $R: T \rightarrow Formulas$ is a well-defined constraining mapping, which associates each transition t in T with a first order logic formula defining the meaning of the transition. Each formula $R(t)$ can be normalized into $Pre-cond(t) \wedge Post-cond(t)$. $Pre-cond(t)$ defines the selection criterion of incoming tokens and $Post-cond(t)$ defines the tokens to be produced.
- (4) $M_0: P \rightarrow Tokens$ is a sort-respecting initial marking. The initial marking assigns a multi-set of tokens to each place p in P .

Thus a HLPN net is $PN = (P, T, F, Spec, \varphi, L, R, M_0)$ and its dynamic semantics is defined by all execution sequences $M_0[T_1/\alpha_1 > M_1[T_2/\alpha_2 > \dots M_n[T_{n+1}/\alpha_{n+1} > \dots]$, in which each T_i is an execution step consisting of a set of non-conflict firing transitions. We have developed a tool environment PIPE+ [8] to create and analyze HLPNs. Our analysis techniques include simulation, model checking (SPIN), bounded model checking (Z3), and term rewriting (Maude). The integrations with external tools currently only support basic functionalities and are being enhanced. The latest source code is available at <https://bitbucket.org/ptnet/pipe>.

B. Security Patterns

Structural patterns contain descriptions of the structure and interactions of security assurance mechanisms, and thus they are suitable for formal modeling and analysis. In this paper, six structural patterns [2] are modeled and analyzed, including *account lockout*, *authenticated session*, *client data storage*, *encrypted storage*, *password authentication*, and *password propagation*. Many of these patterns involve encryption and decryption, session identification, web resource, and timing constraints such as duration and time out. We can only simplify the most of the above entities during the modeling without considering actual encryption and decryption algorithms and session identifier generation. We model a simple clock mechanism to deal with timing constraints. Most of these patterns contain some brief procedural descriptions of user and system interactions complemented with illustrative diagrams. Often the descriptions are not complete and not precise, which are common problems with natural language descriptions. During the modeling process, we need to make hidden assumptions explicit such as the uniqueness of session and user identifications. Although each model can be built with our tool PIPE+ in a few hours after carefully studying the requirements descriptions, it often takes several iterations to get them right, i.e. they capture the requirements correctly to the best of our understanding. However, we have not obtained the approval of the security domain experts.

Due to space limit, we only provide the detailed description of the *authenticated session* security pattern (the most complex pattern with a most complete procedural description among the six). The models of other patterns can be found at <http://users.cis.fiu.edu/~hex/PIPE+.html>.

Authenticated Session Pattern

An authenticated session allows a user to access multiple access-restricted pages on a website only authenticating once on the first page request. The process of authenticated session [2] contains the following steps:

Scenario 1 – Page request without valid authentication

- (1) The client requests a protected page from the server, passing the session identifier;
- (2) The underlying session mechanism retrieves the session data and invokes the protected page object;
- (3) The protected page invokes the authentication checkpoint;
- (4) The authentication checkpoint determines that the authenticated identity field is empty or that the last access time exceeds the session timeout window;
- (5) The requested page URL and submitted data is stored in the session object as the page originally requested;
- (6) The authentication checkpoint redirects the client to the login screen.

Login

The following interactions occur in the login procedure:

- (1) The client requests the login page, submitting a username and password. The session identifier is passed as part of the request;
- (2) The underlying session mechanism retrieves the session data and invokes the login page object;
- (3) The login object validates the username and password;
- (4) If unsuccessful, the unsuccessful login page (“try again”) is returned to the user;
- (5) If successful, the identity provided is stored in the authenticated identity field, and the current time in the last access time;
- (6) The user is redirected to the page originally requested (stored in the session data).

Scenario 2 – Page request with valid authentication

The following interactions occur on a page request with valid authentication:

- (1) The client requests a protected page from the server, passing the session identifier;
- (2) The underlying session mechanism retrieves the session data and invokes the protected page object;
- (3) The protected page invokes the authentication checkpoint;
- (4) The authentication checkpoint validates the authenticated identity field in the session data and

ensures that the last access time does not exceed the timeout period;

- (5) The authentication checkpoint stores the current time as the last access time;
- (6) The authentication checkpoint returns to the protected page object, which composes and delivers the requested page to the client.

Logout

The following interactions occur in the logout procedure:

- (1) The client requests the logout page from the server, passing the session identifier;
- (2) The underlying session mechanism retrieves the session data and invokes the logout object;
- (3) The logout object clears the authenticated user id field in the session;
- (4) The logout object returns a “logged out” notification page or redirects the browser to the home page.

The above procedural descriptions are complemented with an illustrative diagram in Fig. 1:

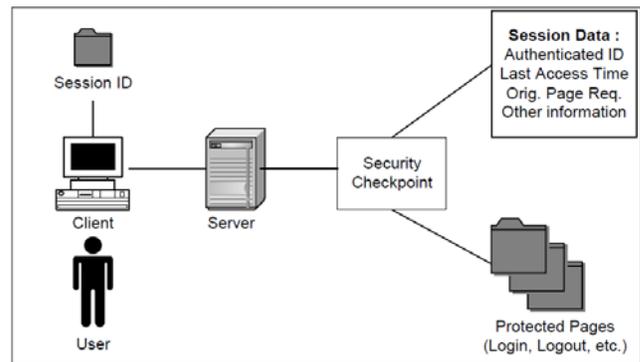


Fig.1 – The Authenticated Session Diagram

HLPN Model of the Authenticated Session Pattern

The HLPN model of the *authenticated session* pattern is shown in Fig. 2, where the type definitions of places and constraints of transitions are provided. For example, a web request is defined as consisting of a session id (*Sid*), and a web page number (*Page#*). The actual types of the above abstract entities are defined as integers. Since there can be multiple requests, a power set type is defined for place *Wreq*. Please note, an arc label connecting to a place of a power set type is represented as $\{x\}$ to indicate the access of a single token. Likewise, a user account (*Acct*) is abstracted as a tuple containing a user name (*Uid*) and a password (*Psw*), both are defined as strings.

The model captures the authentication mechanism with the assumption of the uniqueness of user name, which was implicitly assumed in the original pattern description. Page data are defined as strings and their actual uses are not modeled. In this model, system actions in the authentication process are modeled with transitions. Transition *Retrieve*

captures the steps (1) and (2) in Scenarios 1 and 2. Transition *Check1* models the steps (3) to (6) in Scenario 1. Transitions *AuthF* and *AuthS* define the login activities. Transition *Check2* specifies the steps (3) to (6) in Scenario 2. Transition *Timeout* models step (4) in Scenarios 1 and 2. Transition *Logout* models the logout activities. Transition *Inc* is used to model the increment of clock. *du* is a constant denoting the duration for timeout from the last access. For example, the constraint of transition *Check1* contains the precondition $o[2] = "null"$ or $o[3] + du > c$ (the first time request or expired session).

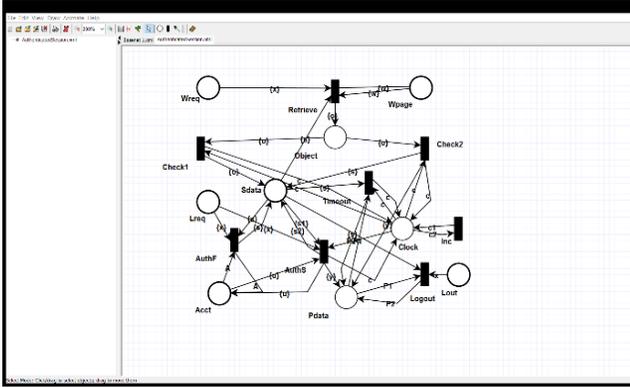


Fig.2 – HLPN model of *Authenticated Session*

$$\begin{aligned}
\varphi(Wreq) &= \wp(Sid \times Page\#), \\
\varphi(Wpage) &= \wp(Page\# \times Data), \\
\varphi(Object) &= \wp(Sid \times Uid \times Last \times Data), \\
\varphi(Sdata) &= \wp(Sid \times Uid \times Last \times Data), \\
\varphi(Lreq) &= \wp(Uid \times Psw \times Sid), \\
\varphi(Acct) &= \wp(Uid \times Psw), \\
\varphi(Clock) &= Time, \\
\varphi(Pdata) &= \wp(Sid \times Data), \\
\varphi(Lout) &= Sid, \\
R(Retrieve) &= s[1] = x[1] \wedge w[1] = x[2] \\
&\quad \wedge o = \langle s[1], s[2], s[3], w[2] \rangle, \\
R(Check1) &= o[2] = "null" \vee o[3] + du \leq c \\
R(Check2) &= (o[2] \neq "null" \wedge o[3] + du > c) \\
&\quad \wedge s = \langle o[1], o[2], c, o[4] \rangle \wedge y[1] = o[1] \\
&\quad \wedge y[2] = o[4], \\
R(AuthF) &= x[3] = s[1] \wedge \exists u \in A. (u[1] = x[1] \wedge u[2] = x[2]), \\
&\quad \wedge msg = "try again", \\
R(AuthS) &= x[1] = u[1] \wedge x[2] = u[2] \wedge x[3] = s[1] \\
&\quad \wedge s1[4] \neq "null" \wedge s2 = \langle s1[1], x[1], c, s1[4] \rangle \\
&\quad \wedge y[1] = s1[1] \wedge y[2] = s1[4], \\
R(Timeout) &= s[3] + du \leq c \\
&\quad \wedge P2 = P1 \setminus \{p \mid \forall p \in P1. (p[1] = s[1])\}, \\
R(Logout) &= x[1] = s[1] \\
&\quad \wedge P2 = P1 \setminus \{p \mid \forall p \in P1. (p[1] = s[1])\}, \\
R(Inc) &= c2 = c1 + 10.
\end{aligned}$$

IV. Analyzing Security Pattern Models

Properties of security patterns are defined using first order linear time temporal logic (LTL) [9], which is an extension of classic first order logic with temporal operators. Predicates in LTL are places and transitions in HLPN. A LTL formula is evaluated under a HLPN model. It should be noted that only correctness properties of the security models are specified and analyzed in this paper. General security properties such as confidentiality, integrity, and availability are not directly studied in this paper, which cannot be specified using temporal logic [10].

A. Specifying Correctness Properties

We have identified and specified three to four correctness properties for all six patterns. In this section, we show how to formulate various correctness related properties in the *authenticated session* pattern. We have identified the following properties:

- The initial web access request to a protected resource must pass authentication. This property is a little tricky to specify. An initial web access implies a session object without an authenticated user identifier (empty field), thus the firing of transition *Check1*. Furthermore the actual access requires the firing of transition *AuthS*:
 $Object(i, "null", *, *) \Rightarrow \diamond Check1(i, "null", *, *, c),$
 $Pdata(i, d) \Rightarrow \diamond AuthS(i, u, *, d, c),$
 where \diamond is a past temporal operator;
- The successive web access requests to a protected resource within the current session must pass session data validation:
 $Object(i, u, c, *) \wedge c + du > clock \Rightarrow$
 $\diamond Check2(i, u, *, *, c);$
- The authenticated session expires after a preset inactive time period:
 $Sdata(i, u, t, d) \wedge c \geq t + du \Rightarrow \diamond \neg Sdata(i, u, t, d)$
- Session data are cleared to prevent an attacker to revisit cached pages:
 $Sdata(i, u, t, d) \wedge Lout(i) \Rightarrow \diamond \neg Sdata(i, u, t, d)$

In the above formulas i, u, c, d, t are free variables, $*$ denotes any legal value. du is a constant. Most of the above properties are liveness properties.

B. Analyzing Properties

Liveness properties in general are very difficult to verify using model checking since they cannot be proved or refuted using finite execution sequences. Furthermore, all the models of the six patterns use power sets of tuples consisting of integers and strings, and some are clearly infinite state models, thus cannot be directly model checked without applying some abstraction techniques to simplify them. Since model checking is not applicable, we seek weaker assurance

through finding finite sequence witnesses through simulation. Thus we turn the verification of the above liveness properties into an easier reachability checking problem. One problem with reachability checking is that it depends on the given initial marking. Fortunately, our reachability checking does not depend on the specific values of the free variables appearing in the formulas as long as certain relationships between these variables are kept. This is similar to the idea of predicate abstraction in model checking, where a generic variable can be replaced with a Boolean variable as long as the correct relationship is maintained. As a result, we can use any randomly generated initial marking satisfying the variable relationships to ensure the validity of the reachability checking. It should be noted that the above properties have been validated but not verified. The simulator has shown that there is a transition firing sequence from an initial marking satisfying any of the above properties; but has not shown that every transition sequence satisfies the above properties. Even these weaker results can be extremely valuable in detecting incomplete and faulty requirements, and eliminating many design errors.

Table 1 summarizes the size of the six models and the number of properties specified and validated.

Table 1 – The Size Metrics of the Models

	Place	Transition	Arc	Property
Account Lockout	6	6	22	4
Auth. Session	9	8	38	4
Client Data Storage	7	6	19	4
Encrypted Storage	7	4	17	4
Password Auth.	6	3	12	3
Password Prop.	12	7	25	4

Table 2 provides the validation results of properties of the authenticated session, which are representative for other patterns. The main parameter is the number of user accounts in the initial markings. The time (ms) is the mean of five runs due to the random firings of enabled transitions such that each run may yield a different running time. Property (a) depends heavily on the user account number and thus takes more time. The experiments are run on Intel® Core(TM) i7-4770S CPU @ 3.10 GHz with 8 GB RAM.

Table 2 – Validation Results

	Token	Time	Token	Time	Token	Time
(a)	10	7	1000	255	10000	38749
(b)	10	1	1000	115	10000	124
(c)	10	3	1000	3	10000	3
(d)	10	0	1000	0	10000	0

V. Concluding Remarks

In this paper, we applied high level Petri nets to formally model and analyze six well-known security patterns.

Building a formal model from given textual descriptions is often quite difficult due to the incompleteness and ambiguity of such descriptions, and thus error prone. Identifying and correctly specifying relevant properties is another major challenge. Formally verifying whether a formal model satisfying the specified properties is very hard due to the complexity of the models that often results in state explosion problem. We have shown the usefulness of reachability analysis based on simulation as a supplemental validation technique. The validated security patterns form the basis for composing multiple security patterns as well as integrating security patterns with other system components.

Acknowledgements

This work was partially supported by the NSF under grant HRD-0833093 and by the AFRL under agreement number FA8750-15-2-0106. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon.

References

- [1] N. Yoshioka, H. Washizaki, and K. Maruyama: “A survey of security patterns”, *Progress in Informatics*, no. 5, 35-47, 2008.
- [2] D. M. Kienzle, M. C. Elder, D. Tyree, and J. Edwards-Hewitt: “Security Patterns Repository Version 1.0”, 2006.
- [3] S. Konrad, B. Cheng, L. Campbell, and R. Wassermann: “Using Security Patterns to Model and Analyze Security Requirements”, *International Workshop on Requirements of High Assurance Systems*, 2003.
- [4] W. E. McUumber and B. H. C. Cheng: “A general framework for formalizing UML with formal languages”, *Proceedings of IEEE International Conference on Software Engineering*, Toronto, Canada, May 2001.
- [5] A. Dwivedi and S. Rath: “Formalization of Web Security Patterns”, *INFOCOMP*, v. 14, no. 1, p. 14-25, June 2015.
- [6] J. Dong, T. Peng, and Y. Zhao: “Automated verification of security pattern compositions”, *Information and Software Technology*, vol. 52, 2010, 274-295.
- [7] X. He: “A Comprehensive Survey of Petri Net Modeling in Software Engineering”, *International Journal of Software Engineering and Knowledge Engineering*, vol. 23, no. 5, 2013, 589-626.
- [8] S. Liu, R. Zeng, X. He: “PIPE+ - A Modeling Tool for High Level Petri Nets”, *Proc. of International Conference on Software Engineering and Knowledge Engineering*, Miami, July 2011, 115 - 121.
- [9] Z. Manna and A. Pnueli: “*The Temporal Logic of Reactive and Concurrent Systems – Specification*” Springer-Verlag, Berlin, 1992.
- [10] M. Clarkson and F. Schneider: “Hyperproperties”, *Journal of Computer Security*, vol. 18 (2010) 1157-1210.

Evaluating the Representation of User Interface Elements in Feature Models: an Empirical Study

Ildevana Poltronieri Rodrigues*, Ana Paula Terra Bacelo*, Milene Selbach Silveira*,
Márcia de Borba Campos*, Elder de Macedo Rodrigues†

*School of Computer Science - Pontifical Catholic University of Rio Grande do Sul (PUCRS) – Porto Alegre, Brazil

†Federal University of Pampa – Alegrete, Brazil

Email: ildevana.rodrigues@acad.pucrs.br {ana.bacelo, milene.silveira, marcia.campos}@pucrs.br
eldermr@gmail.com

Abstract—Feature models are considered a reference in the majority of Software Product Line (SPL) methods. There are several different feature model notations to represent requirements of an SPL, at a higher or lower level of abstraction. Some notations present properties to distinguish functional, conceptual and non-functional features of an SPL. Non-functional requirements, especially those that involve the construction of user interfaces (UI), are usually not represented in the feature models, since the user interfaces are often created manually for each product. In this paper we present an experimental study performed in order to evaluate the effort required to, as well as the benefits and drawbacks of representing UI elements during the feature modeling of a Financial SPL. To this end, we applied the Odyssey-Fex and UI-Odyssey-Fex notations to design feature models from the perspective of Domain Engineers in the context of undergraduates, M.Sc. and Ph.D. students and software engineers with some expertise in SPL. Our results indicate that the effort to use the notations are similar, but the use of the UI-Odyssey-Fex notation provides a better representativeness of UI elements.

I. INTRODUCTION

A feature model (FM) represents all of the features, and the relationships between them, of a software product line (SPL) domain [1] [2] [5] [6]. A feature can be mandatory (must be present in all of the derived products) or optional (one or more domain products have this feature). A derived product is composed by the mandatory features and a selection of optional features.

There are different feature model notations, with an assorted number of elements, to represent the features of the SPL domain at different levels of abstraction [7]. Some notations present properties to distinguish functional, conceptual and non-functional features of an SPL. However, beyond these, in the construction process of a feature model for an SPL, user interface elements can be modeled in order to identify mandatory, optional and configurable interfaces among the different products of an SPL. Although there are several researches dealing with non-functional properties [13], those involved in the construction of user interfaces are normally not represented in the feature models or are not addressed in the context of the construction of an SPL. Pleuss et al. [11] point out that, in practice, the user interfaces are more often than not

created manually for each product, which impacts the usability, efficiency and maintenance of the product, without considering the aspects of cost involved in their construction.

There are different approaches for including usability aspects in software product lines, from their manual construction, as previously mentioned, to extending existing methods for software product lines, to include non-functional requisites [9] in possibilities of customization based on abstract models [4]. Using abstract models and even extending already existing models is very costly, with the team having to develop (and/or extend) one more of many models, which ends up being unfeasible considering issues regarding the necessary cost, time and human resources.

Although there are many FM modeling notations available, and a few of which supporting UI elements, SPL engineers face some doubts when considering to model UI elements into a FM, such as: Does an UI-supporting notation present some advantage regarding ease of use when compared with a standard FM notation?; What is the effort required to create a FM with UI elements? Does the use of an UI-supporting notation provide a better representativeness of UI elements?. Despite these doubts, to the best of our knowledge there is no work discussing the required modeling effort, benefits and issues on the use of notations that support the representation of UI elements. Motivated by this lack of knowledge, in this paper we report an experimental study to provide evidence about the advantage, the required effort to design the models and representativeness of UI elements.

This paper is organized as follows. Section 2 presents the experiment instruments, the FM notations, the SPL requirements documents and our design guidelines to model the feature diagrams. Section 3 provides details about the experimental design and Section 4 describes the execution of the experiment. Section 5 presents our analysis on achieved results. We conclude the paper in Section 6 with final considerations and some future research directions.

II. EXPERIMENT INSTRUMENTS

In this section we briefly introduce the Odyssey-Fex and UI-Odyssey-Fex feature model notations.

A. Odyssey-Fex

The Odyssey-Fex notation [3] is used in the Odyssey reuse environment [10]. Different from the notations proposed by

Favaro and Mazzini [4], Gulp et al. [15] and Czarnecki et al. [2], this notation also indicates that the features should be classified into different types: Domain, Entity, Operational, Domain Technology and Implementation Techniques. The Odyssey-Fex notation also proposes a different way of representing the relationship between features. As such, it uses the representation of UML relationships and the specific notation relationships, as proposed by Miller [8]. In addition to the different types of features and their relationships, the variability aspects of an SPL in the OdysseyFex notation are also considered. As such, features can be specified as mandatory (they exist in all of the products of an SPL), optional (they may or may not exist in one or more of the products of a line) or variation points (they are configurable considering the product of which it is part). Figure 1 shows an example of feature model created in accordance with the Odyssey-Fex notation, for the domain of Financial Institutions. The Financial domain is represented by the conceptual feature *Financial*, with which the actors *Customer* and *Employee* interact. This feature is associated with the functional feature *Authenticate*. The *Authenticate* feature is implemented by several other functional features, such as *Credit Card Balance* and *Transfer*. The mandatory *Transfer* feature must be implemented by the *Checking to Saving*, *Saving to Checking*, *Accounts from Others Banks* and *Accounts from Same Bank*. As can be seen in the figure, there is no element of this notation to explicitly represent UI components in the FM.

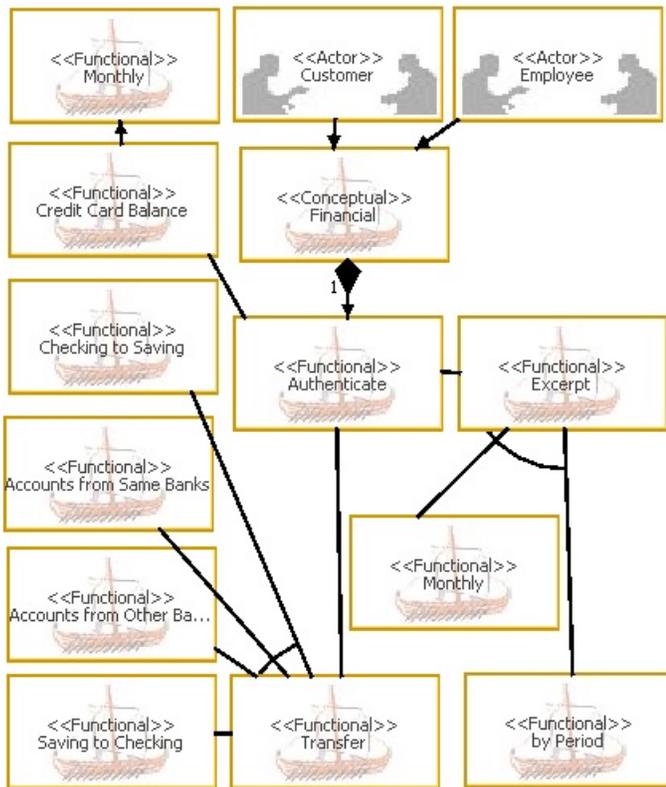


Fig. 1. Feature Model – Odyssey-Fex notation

B. UI-Odyssey-Fex

To mitigate the Odyssey-Fex limitation in representing UI elements an extension, called UI-Odyssey-Fex, was developed.

This extension provide a feature element to model features that represent aspects of user interface [11]. In UI-Odyssey-Fex, all elements need to represent features in the original notation were implemented and, in addition to this, a new feature category was included, called User Interface. An example of the FM designed in accordance with the extension is presented in Figure 2. The figure shows the same SPL represented in Section II-A, but including the User Interface features category. In this case, the SPL engineer could represent the UI elements that are shared among the Financial products derived from the SPL. For instance, the *Transfer* functional feature has a communication link with the User Interface *Transfer System* feature.

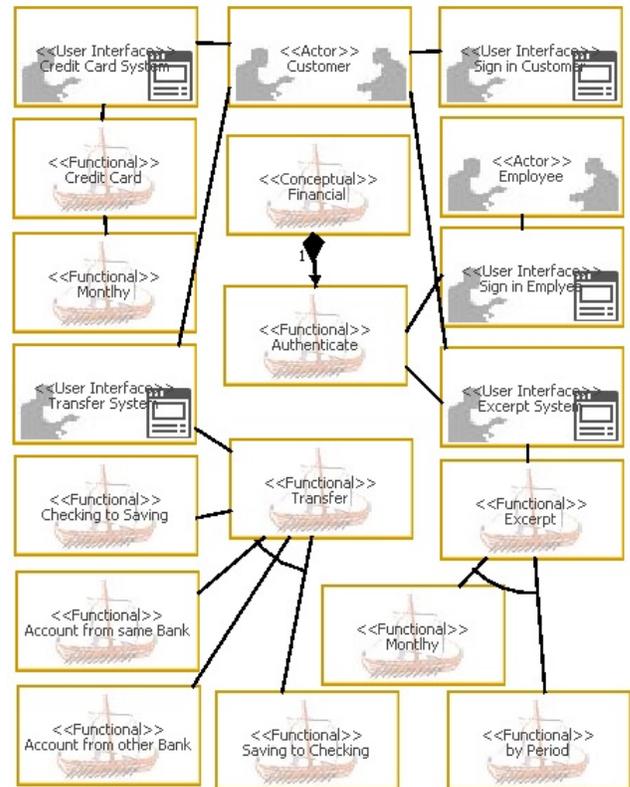


Fig. 2. Feature Model – UI-Odyssey-FEX notation

III. EXPERIMENT DESIGN

In this section we introduce the Research Questions (RQ), present our experimental design, describe the experiment instruments and discuss the main threats to the study validity of the study:

RQ1: Does the UI-Odyssey-Fex notation present advantages regarding ease of use, when compared with the Odyssey-Fex notation?

Null Hypothesis H_0 : There is no advantage, regarding ease of use, when using Odyssey-Fex or UI-Odyssey-Fex notations. **Alternative Hypothesis H_1 :** The use of UI-Odyssey-Fex notation presents some advantage regarding ease of use when compared with the Odyssey-Fex notation. **Alternative Hypothesis H_2 :** The use of Odyssey-Fex notation presents some advantage regarding ease of use when compared with the UI-Odyssey-Fex notation.

RQ2: Does the UI-Odyssey-Fex notation require less effort (time spent) to design a feature model, when compared with the Odyssey-Fex notation?

Null Hypothesis H_0 : The effort is the same when using Odyssey-Fex or UI-Odyssey-Fex notations to design a feature model. **Alternative Hypothesis H_1 :** The use of the UI-Odyssey-Fex notation requires less effort to design a feature model, when compared with the Odyssey-Fex notation. **Alternative Hypothesis H_2 :** The use of the Odyssey-Fex notation requires less effort to design a feature model, when compared with the UI-Odyssey-Fex notation.

RQ3: Does the UI-Odyssey-Fex provide a better representativeness of UI elements, when compared with the Odyssey-Fex notation?

Null Hypothesis H_0 : The representativeness is the same, when using the Odyssey-Fex or UI-Odyssey-Fex notations, to represent UI elements. **Alternative Hypothesis H_1 :** The use of the UI-Odyssey-Fex notation better represents UI elements when compared with the Odyssey-Fex notation. **Alternative Hypothesis H_2 :** The use of the Odyssey-Fex notation better represents UI elements when compared with the UI-Odyssey-Fex notation.

A. Design

In our experiment we used an in-vitro approach, since all the experiment sessions were executed in a laboratory, under controlled conditions and using documentation from a toy example SPL. As subjects, we invited undergraduate, master and Ph.D. students from a university; from a company we invited software engineers with some expertise in SPL. The students were from Computer Science and Information Systems courses. It is important to highlight that each student had a different level of knowledge on SPL concepts, FM design and UI element representation. For instance, some students had some experience as a software analyst or as a developer, or experience designing and developing software during an IT course. The professional subjects were from an IT company, with an advanced level of knowledge about SPL modeling, but with an introductory knowledge on UI concepts. It is important to note that, initially, all the subjects answered a survey, thus enabling us to classify them into three blocks (beginner, intermediate and advanced), according to their background in software engineering - such as software design with UML and software product line concepts (see Figure 3). After that, the subjects were randomly (randomized block design [16]) allocated in two groups: Group 1 and Group 2. Subjects in Group 1 started modeling a SPL using the UI-Odyssey-Fex notation and then using the Odyssey-Fex notation to create an equivalent model. Conversely, Group 2 started modeling a SPL using the Odyssey-Fex notation and then applied the UI-Odyssey-Fex notation to create an equivalent model. Hence, all the subjects executed both treatments (Odyssey-Fex and UI-Odyssey-Fex notations). Moreover, we also *balance* the groups, so a similar number of subjects started modeling a feature model with a different notation (Group 1 with 16 subjects and Group 2 with 15 subjects). Figure 3 summarizes the experiment phases.

B. Instrumentation

During the experiment design and execution phases several *objects* were generated/used. The main experiment objects are the FM diagrams designed by the subjects to model a financial SPL - one diagram for each notation. Moreover, other objects were prepared and provided to support the subjects during the execution of the experiments tasks, such as training material, applications requirements, tools manual and examples of feature model diagrams. To create the features models we have provided a modeling supporting tool, called Odyssey [10]. This tool and the notations were presented to the experiment subjects during the training sessions, where we provided a manual with detailed instructions on how to apply the modelling tools to design the feature models in accordance with each notation. These sessions were conducted in a controlled laboratory, and all the experiment's subjects attended simultaneously. During the training sessions, the subjects modeled an SPL for the Retail domain, and they could ask open questions about the application documentation, notations and tools. These questions and their answers were shared among all the subjects. In the experiment execution phase the subjects modeled, using both modeling notations, an SPL for the Financial domain, based on its documentation. The modeling tool used were the same and all the subjects used the same computational resources. Moreover, the laboratory was isolated to avoid interruptions and any communications among the subjects or with the exterior world (e-mail, instant message, cellphones, etc) were not allowed. During the execution we collected effort metrics for each subject, as well as collected qualitative data after every session through a questionnaire.

C. Threats to Validity

Throughout the experimental process we identified some threats to the validity of our experimental study. In this section we describe these threats, as well as how we worked to mitigate them. To classify the threats we adopted the classification scheme proposed by [16]:

- **Conclusion validity:** we had identified some threats that could affect the ability to draw conclusions about relations between the treatment (notations) and the outcome of our experiment. One of the major threats is the small sample of subject (31 subjects), and the heterogeneity of the subjects profiles. They had a different level of expertise on software engineering, software product lines and modeling, and their different academic degrees. To mitigate these threats we took into consideration their academic degree and professional expertise to categorize them into blocking variables (beginner, intermediate and advanced).
- **Internal validity:** to mitigate the threats to the internal validity of the experiment we took the following decisions during the experiment design: the experiment session (training and execution) dates were defined to avoid periods in which subjects may be exposed to external influences, e.g., avoid running the experiment during the exam period (student subjects) and close to the start/end of a subject's project (industry subjects).

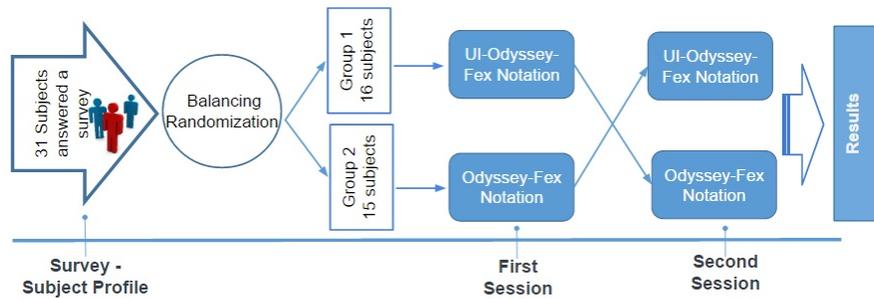


Fig. 3. Experiment Phases

- External validity:** we were aware that the selected sample of subjects may be a threat to the external validity of our study, since those subjects may not represent the SPL modeling professionals (population). To mitigate this threat, we only selected students and professionals with skills/expertise in this domain, such as SPL concepts and software modeling. Another threat to our study was that since the subjects learn the Odyssey-Fex notation, it could be easier to model with UI-Odyssey-Fex (it is an extension of Odyssey-Fex notation). This learning curve may influence the results. To mitigate this threat, the subjects of each block were randomly divided into two groups, where each group started modeling with a different notation. Another threat is that the application documentation may not be clear and unambiguous, leading the subject to misunderstand the SPL domain, its features and their relations, as well as its UI elements. To mitigate this threat we invited an expert in software product lines, software reuse and modeling, and another expert in Human Computer Interaction to review the experiment artifacts.
- Construct validity:** it may occur that a subject can erroneously conclude that their personal performance is measured to rank the experiment subjects or decide to respond favorably to the researcher. For instance, a student may be inclined to answer positively a survey of his professor. To mitigate these threats, before each session a researcher explained that the focus of the study was to evaluate modeling notations, not the subjects. Moreover, we clearly stated that we were not involved in the development of any of the notations, so we do not have any preference. The correct analysis and statistical interpretation of the results may be another threat to experiment construct validity. To mitigate this threat we invited an expert in statistical analysis from the Faculty of Mathematics of PUCRS.

IV. OPERATION OF THE EXPERIMENTAL STUDY

In this section we present in detail the preparation and execution activities performed during the experiment operation.

A. Preparation

In the preparation, we present how the experiment was conducted, how the documentation was prepared and how the experiment environment was configured. We also present how the experiment subjects were involved and motivated.

Our first personal contact with the subjects was made through an introductory presentation session, where an initial explanation

about the general idea of the experiment was presented. At this first moment we also provided a time slot, where the subjects could ask open questions about the experiment setup as well as about SPL and the FM notations and tools. At this moment we also discussed the research goal and how the subjects' data (e.g. experiment effort data) would be published. During this first session we presented the basics of software product lines, FODA [5] and FeatuRSEB [15] methods, as well as the Odyssey-Fex and UI-Odyssey-Fex notations.

TABLE I. EXPERIMENT SUBJECT DISTRIBUTION

Groups	Blocking		Number of subjects
Group 1	Students	Beginner	6
		Intermediate	2
		Advanced	1
	Industry	Beginner	5
		Intermediate	2
		Advanced	0
Group 2	Students	Beginner	3
		Intermediate	4
		Advanced	1
	Industry	Beginner	6
		Intermediate	1
		Advanced	0
Total of Subjects			31

During the second session, performed in a different date, manuals were presented to guide the use of the Odyssey-Fex and UI-Odyssey-Fex notations and tools. At this time, all the participants received instructions on the semantics of models, as well as a detailed demonstration of how to use the tools. At the final part of the session it was presented an example of a requirements document of a retail SPL (similar to a MD-50 functional requirement document) and its correspondent feature model. Based on this, it was possible to discuss and answer questions from participants on the notations' syntax, such as on the use of tools. At the end of the session a subject profile form was provided to all the subjects. The information extracted from the forms was used to classify and distribute the subjects through the blocks before running the experiment. To classify a subject into the blocks we used an ordinal scale for each response of a multiple choice questionnaire [16]. For instance, the subjects' experience with SPL concepts and feature modeling design was measured on a four level ordinal scale: 1 = none, 2 = beginning level (studied in a class), 3 = intermediate level (practised in a class project or in a small industry project), 4 = advanced level (used in more than one industrial project). Based on this, we evaluated the average experience of each subject and used it to distribute them into the blocks. The distribution of the industrial and student subjects into the blocks is depicted in Table I.

B. Execution

The experiment execution phase was comprised of two sessions (blue boxes in Figure 3): Session one - was used by the subjects from the Group 1 to design the FM using the UI-Odyssey-Fex notation, while subjects from Group 2 designed the FM using the Odyssey-Fex notation. It is important to note that to avoid misunderstanding, we provided the requirement documents with only those requirements, seven requirements in total, that the subject must design. Both groups used the same application documents: requirements and models. Moreover, they also designed the models at the same time, but they used different and isolated lab rooms. Session two - was used by the Group 1 subjects to design the FM using the Odyssey-Fex notation, while subjects from Group 2 designed the FM using the UI-Odyssey-Fex notation. As in the first session, both groups, separately, applied the same application documents that had been provided previously in session one. It is important to highlight the sessions one and two were performed in different dates. To monitor the time spent by each subject, the researcher who ran the experiment used a software to control the starting and ending time of each subject.

At the end of each session, all the subjects, from both groups, answered the same survey, to evaluate the ease of use and representativeness of each notation. For each survey question we provided three close-ended options (Linkert scale): *Disagree*, *Neither agree nor disagree* and *Agree*.

V. RESULTS

In this section, we present the data collected from our experiment, as well as summarizing our general findings. We performed hypothesis testing for all data sets using the R Studio statistical environment for R [12]. First, we evaluated the influence of the order in which the subjects applied the treatments (group one or group two), with the Chi-Square (χ^2) normality test [14], considering a significance level $\alpha = 0.05$. The results for the data collected from both notations shows that the difference between the observed and expected frequencies in each category are very small (see Table II), allowing us to reject the assumption that the order in which the subjects apply the notation did not affect the results.

TABLE II. CHI-SQUARE NORMALITY TEST

Treatments	χ^2	α
UI-Odyssey-Fex	1.9273	0.4658
Odyssey-Fex	2.6467	0.3233

A. Hypothesis Testing

Since the order in which the subjects applied the treatments does not influence the results, we applied parametric Student's t-test for paired samples, considering a significance level $\alpha = 0.05$ and 30 degrees of freedom. In the remainder, we present and discuss the results per survey question, which in turn are related to the RQs stated in Section III. To evaluate the subjects' answers, each survey question has a ordinal scale for each response [16]. Therefore, the subjects must choose an answer that is measured on a three level ordinal scale: (1) agree, (0) partially agree and (-1) disagree. We also performed hypothesis testing applying t-Test (for paired samples), considering a significance level $\alpha = 0.05$ and 30 degree of freedom.

TABLE III. SUBJECTS ANSWERS - RQ 1

Subjects Answers								
Scale	Odyssey-Fex				UI-Odyssey-Fex			
	Group 1	Group 2	Total	%	Group 1	Group 2	Total	%
Agree	8	4	12	38.71	7	8	15	48.39
Partially Agree	5	9	14	45.16	5	6	11	35.48
Disagree	3	2	5	16.13	4	1	5	16.13

RQ1: *Does the UI-Odyssey-Fex notation present advantages regarding ease of use, when compared with the Odyssey-Fex notation?*

This RQ is represented by a closed multiple choice question in both subjects surveys: *Does the (UI-Odyssey-Fex/UI-Odyssey-Fex) notation present advantages regarding ease of use?*. As shown in Table III, 38.71% of the subjects agreed that the Odyssey-Fex is easier to use, while 45.16% partially agree and 16.13% disagree with the affirmation. In turn, when design the models with the UI-Odyssey-Fex, 48.39% of the subjects agreed that the notation is easier to use and 16.13% disagree. This basic analysis indicate that UI-Odyssey-Fex is easier to use than Odyssey-Fex. However, we performed hypothesis testing in order to accept or reject the H_0 hypothesis. For the data set from the question we applied the t-Test to the paired sample, in this way comparing the answers for Odyssey-Fex and UI-Odyssey-Fex. The results are shown in Table IV. The results of the test was **t-Test = 0,8278**, which is smaller than the tabulated value (from the t-table [16]) which is **t= 2,0423**. Therefore, we **accept** the H_0 hypothesis and conclude that there are no differences, regarding ease of use, when using Odyssey-Fex or UI-Odyssey-Fex notations.

TABLE IV. PAIRED T-TEST QUESTION 1 - EASINESS OF USE

Treatments	Paired t-Test	Degrees of Freedom	p-value
Odyssey-Fex e UI-Odyssey-Fex	0.8278	30	0.4143
Confidence Intervals 95%	- 0.1419 - 0.3355		
Mean of difference	0.0967		

RQ2: *Does the UI-Odyssey-Fex notation require less effort (time spent) to design a feature model, when compared with the Odyssey-Fex notation?*

To answer this RQ we collected the subjects' effort data, in minutes, per notation. Based on this data, we applied the t-Test for paired samples. As Table V shows, the result of the test was t-Test = -1.1803, which is smaller than the tabulated value of t= 2.0423 and a p-value of 0.2471 (which is greater than 0.05). Based on that we can **accept** the H_0 hypothesis and conclude that the effort is similar when using the Odyssey-Fex or UI-Odyssey-Fex notations to design a feature model.

TABLE V. PAIRED T-TEST - EFFORT DATA

Treatments	Paired t-Test	Degrees of Freedom	p-value
Odyssey-Fex e UI-Odyssey-Fex	-1.1803	30	0.2471
Confidence Intervals 95%	-9.8641 - 2.6383		
Mean of difference	-3.6129		

RQ3: *Does the UI-Odyssey-Fex provide a better representativeness of UI elements, when compared with the Odyssey-Fex notation?*

To answer this RQ we collected the data from two of the evaluation survey questions that refer to the syntax and semantics of UI elements represented in the notations. Based on this data, we applied the t-Test for paired samples. As Table VI shows, regarding to the syntax question, the result of the test was $t\text{-Test} = 3.8676$, which is higher than the tabulated value of $t=2.0423$ and $p\text{-value}$ of 0.0005, that is, significantly lower than the expected 0.05. Based on that we can **refute** the H_0 hypothesis and **accept** the H_1 hypothesis, concluding that the UI-Odyssey-Fex notation has adequate syntax to represent UI elements when compared with the Odyssey-Fex notation.

TABLE VI. PAIRED T-TEST QUESTION 3 - SINTAXE

Treatments	Paired t-Test	Degrees of Freedom	p-value
Odyssey-Fex e UI-Odyssey-Fex	3.8676	30	0.0005
Confidence Intervals 95%	2.8926 - 0.9365		
Mean of difference	0.6129		

Afterwards, in what refers to semantics, we obtained $t\text{-Test} = 3.1526$ and a $p\text{-value} = 0.0036$, as shown on Table VII, that is, the t value is higher than the tabulated 2.0423, and its $p\text{-value}$ is significantly lower than 0.05, from which we can **reject** the H_0 hypothesis and **accept** the H_1 hypothesis, concluding that the UI-Odyssey-Fex notation has adequate semantics to represent UI elements when compared with the Odyssey-Fex notation.

TABLE VII. PAIRED T-TEST QUESTION 3 - SEMANTIC

Treatments	Paired t-Test	Degrees of Freedom	p-value
Odyssey-Fex e UI-Odyssey-Fex	3.1526	30	0.0036
Confidence Intervals 95%	0.2044 - 0.9679		
Mean of difference	0.5806		

VI. FINAL CONSIDERATIONS

Requirements of a Software Product Line (SPL) are represented with Feature Models. There are several notations for this representation [6] [2] [15] [1] [3], which have distinct properties, though all have the means to represent mandatory and optional features of a system family within an SPL. Conversely, we have observed that some notations do not cover the representation of some non-functional aspects of an SPL, specially related to User Interface (UI) elements.

To overcome this limitation, this research has analyzed the representation of UI features in the building of Feature Models (FM) in SPL, comparing the notation Odyssey-Fex with its extension, UI-Odyssey-Fex. These notations were evaluated based on three Research Questions (RQ), each of which meant to evaluate advantages regarding ease-of-use, effort required and syntactic and semantic representativeness, respectively. The results show that regarding ease-of-use and required effort, there is no statistically significant difference between the two notations. However, there is a significant advantage to representativeness in the use of UI-Odyssey-Fex, when compared to Odyssey-Fex.

We have identified two major threats to the experiment. The first is the size of the sample, as well as its being comprised of a group that may not be representative of the population. The second is the fact that the experiment only compares two

notations, whereas ideally a comparison with other common notations, such as FODA, would be desirable. In future works, we hope to replicate this study with a different sample and more notations, in order to provide a more strong evidences about the benefits and drawbacks of representing UI elements using different FM notations.

REFERENCES

- [1] Q. Boucher, G. Perrouin, and P. Heymans. Deriving configuration interfaces from feature models: A vision paper. In *Proceedings of the Sixth International Workshop on Variability Modeling of Software-Intensive Systems*, pages 37–44, New York, NY, USA, 2012. ACM.
- [2] K. Czarnecki, S. Helsen, and U. Eisenecker. Staged configuration using feature models. In R. Nord, editor, *Software Product Lines*, volume 3154 of *Lecture Notes in Computer Science*, pages 266–283. Springer Berlin Heidelberg, 2004.
- [3] R. F. de Oliveira. Formalization and consistency checking in variabilities modeling. Master thesis, Federal University of Rio de Janeiro, Rio de Janeiro, Brazil, 2006.
- [4] J. Favaro and S. Mazzini. Extending featurseb with concepts from systems engineering. In S. Edwards and G. Kulczycki, editors, *Formal Foundations of Reuse and Domain Engineering*, volume 5791 of *Lecture Notes in Computer Science*, pages 41–50. Springer Berlin Heidelberg, 2009.
- [5] K. Kang, S. Cohen, J. Hess, W. Nowak, and S. Peterson. Feature-oriented domain analysis (FODA) feasibility study. Technical report, Software Engineering Institute, Carnegie Mellon University Pittsburgh, PA., 1990.
- [6] K. C. Kang, S. Kim, J. Lee, K. Kim, E. Shin, and M. Huh. Form: A feature-oriented reuse method with domain-specific reference architectures. *Ann. Softw. Eng.*, 5:143–168, Jan. 1998.
- [7] G. B. Klaus Pohl and F. van der Linden. *Software Product Lines Engineering: Foundations, Principles, and Techniques*. Springer - Verlag Berlin Heidelberg, KünkleLopka, Heindelberg, UE, 2010.
- [8] N. J. Miler. *A Engenharia de Aplicações no Contexto da Reutilização Baseada em Modelos de Domínio*. PhD thesis, Federal Univeristy do Rio de Janeiro, Rio de Janeiro, Brazil, 2000.
- [9] Q. L. Nguyen. Non-functional requirements analysis modeling for software product lines. In *Modeling in Software Engineering, 2009. MISE '09. ICSE Workshop on*, pages 56–61, May 2009.
- [10] L. of Software Engineering COPPE UFRJ. Odyssey project, Setembro 2014.
- [11] A. Pleuss, B. Hauptmann, D. Dhungana, and G. Botterweck. User interface engineering for software product lines: The dilemma between automation and usability. In *Proceedings of the 4th ACM SIGCHI Symposium on Engineering Interactive Computing Systems, EICS '12*, pages 25–34, New York, NY, USA, 2012. ACM.
- [12] R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2015.
- [13] L. R. Soares, P. Potena, I. d. C. Machado, I. Crnkovic, and E. S. d. Almeida. Analysis of non-functional properties in software product lines: A systematic review. In *Proceedings of the 2014 40th EUROMICRO Conference on Software Engineering and Advanced Applications, SEAA '14*, pages 328–335, Washington, DC, USA, 2014. IEEE Computer Society.
- [14] J. Tukey. *Exploratory Data Analysis*. Addison-Wesley series in behavioral science. Addison-Wesley Publishing Company, 1977.
- [15] J. van Gurp, J. Bosch, and M. Svahnberg. On the notion of variability in software product lines. In *Proceedings. Working IEEE/IFIP Conference on Software Architecture, 2001*, pages 45–55, August 2001.
- [16] C. Wohlin, P. Runeson, M. Höst, M. C. Ohlsson, B. Regnell, and A. Wesslén. *Experimentation in Software Engineering: An Introduction*. Springer - Verlag Berlin Heidelberg, Norwell, MA, USA, 2012.

Towards a Systematic Approach to Graph Data Modeling: Scenario-based Design and Experiences

Mengjia ZHAO*, Yan LIU†, Peng ZHOU‡
School of Software Engineering, Tongji University
Shanghai, China

Email: *1434319@tongji.edu.cn, †yanliu.sse@tongji.edu.cn, ‡1435855@tongji.edu.cn

Abstract—Graph database is recently being adopted by data analytic systems as an appealing alternative to relational database for the management of large-scale inherent graph-like data. A great challenge of leveraging graph database technologies is to model a problem domain into graph. However, in the absence of considering application requirements or goals, current graph data modeling approaches seem to be invalid. This paper presents an exploration of a systematic approach for graph data modeling—SuMo. Starting from real world scenarios, requirements are transformed into a domain model, which acts as an intermediate model in SuMo, captures modeling features of that domain. SuMo defines a set of rules for the subsequent transformation of this domain model to produce a graph model. We applied SuMo to the modeling of a data-intensive analytic system using real datasets as an illustrating example to clarify our main idea. SuMo is empirically evaluated in terms of query performance, the experimental results indicate promising feasibility and efficiency. The major contribution of our work is a preliminary graph data modeling approach based on scenarios.

Keywords—graph data modeling; data analytic system; scenario-based modeling; model-driven design

I. INTRODUCTION

Graph database has recently gained popularity rapidly because of a need to effectively manage large-scale inherent graph-like data [1]. Social network, biology, semantic web and health-care are typical application domains that contain such kind of data [2][3][4]. It has been observed that graph database is usually preferable to relational database in managing data of these domains since the latter hardly captures the inherent graph structure [5]. Motivated by scalability and performance needs of current applications, graph database is increasingly adopted by data analytic systems.

The analysis of relationships becomes important when dealing with highly connected data. A graph consists of a finite set of nodes, and a finite set of edges defining relationships between these nodes. In graph database, data is stored as graph and is accessed by queries as graph traversal operations. Compared with the stores of relational database, relationships are treated as first-class citizens in graph database [6], expressed in a more straightforward way. Moreover, queries involving complex and inefficient join operations are transformed into graph traversals. The execution time of graph traversal is proportional only to the size of traversed part [6].

Graph data modeling is the process in which an arbitrary application domain described as a connected graph of nodes and edges, it is not a context-free process, but a purposive abstraction related with application requirements. As it happens with relational database, the design of relational model can start from ER model, such kind of “springboard” is also conducive to the design of graph model.

Guidelines and principles on graph data modeling can be found from various sources, including books, technical reports, graph database online community and practitioners’ blogs. There has also been an involvement of academic papers in the study of graph data modeling, many works focus on converting existing data from relational to graph model automatically; a number of researchers aim at generic graph data modeling approaches. However, most of these works, simply demonstrated some rudimentary strategies of graph data modeling, rather than connecting to practical application requirements. As a result, practitioners can only find some scattered modeling guidelines.

Currently, graph data modeling is still based on best practices and probably unproved guidelines, which are usually relevant to specific systems [5]. This paper aims to propose a systematic graph data modeling approach at a “proof of concept” stage. The primary challenge that come is, how to analyze requirements of application domain to support graph data modeling? Translating these requirements to produce a model brings more challenges.

In this paper, we are exploring a systematic graph data modeling approach, using scenarios to start out the modeling process. We suggest an adapted domain modeling approach with strategies to abstract key concepts from scenarios. A set of rules are defined for the transformation of domain model to graph model. Our aim is to match application requirements and facilitate the design of graph-based analytic systems. We also provide experiments, in terms of query performance, showing the advantages of our approach with respect to a naïve approach. The main contribution of this paper is a sketch of a systematic graph data modeling approach.

The rest of this paper is organized as follows, Section II presents related work. The proposed approach is presented in Section III. A case study is demonstrated in Section IV. We design and conduct a group of experiments to evaluate our approach, as well as discuss the results in Section V. The conclusion and future works are in Section VI.

II. RELATED WORK

Graph database practitioners and enthusiasts have built graph data models in their respective domains [7]. After familiarization of the application domain, they create a basic skeleton of graph model directly and intuitively. Generally, these models need further refinements. Many best practices have emerged in domains of real-time recommendations, fraud detection, social network, etc [8]. Guidelines for graph data modeling can be concluded from these use cases. However, a systematic solution can not be generalized or summarized.

Relational database has been around for many decades and is the choice for most traditional data-intensive storage and retrieval systems. To meet new demands of current applications, a lot of existing systems choose to migrate data layer from relational database to a graph-based storage system. In [9], a methodology was proposed to convert a relational model to a graph model by exploiting the schema and constraints. [10] and [11] focus on mapping relational to graph model without semantically loss, use *Primary Keys* and *Foreign Keys* to create edges between nodes.

A number of researchers have tried to propose generic graph data modeling approaches. In [5], a model-driven methodology was proposed for the design of graph database. It starts from Entity-Relationship (ER) model, translates this conceptual model into graph model following a specific strategy, aims to minimize the number of needed access operations in retrieving data. In our opinion, some purposeful works, like identifying application goals or requirements, are not involved in [5].

Related work so far can not fully support the modeling process of an arbitrary graph-based analytic system from the very start. Our approach tends to concern more about requirements. We will present our work in the following sections to explore a systematic, requirement driven approach for graph data modeling.

III. SUMO: A SCENARIO-BASED APPROACH FOR GRAPH DATA MODELING

We name our approach “SuMo” referring to graph data modeling using scenarios. Fig. 1 presents an overview of SuMo. The modeling processes can be organized in the following two phases:

A. Phase 1: Scenario-based domain modeling

The main difference between what we do in this phase and general domain modeling is that, we are “graph-oriented”. That is to say, every activity we do is purposeful, we set up a graph in mind at the beginning stage. *Discover* and *Invention*, which are used often in OOD, will abstract domain concepts or attributes at early stage. Specifically, constraints derived from scenarios will check the domain entities and their relationships. In the refinement of domain concepts, *Integration* and *Split-off*, together with the former two strategies might be used.

1) *Discovery*: Discovery is a basic strategy to recognize the abstractions used by domain experts. If the scenarios mention it a lot, the abstraction is usually important.

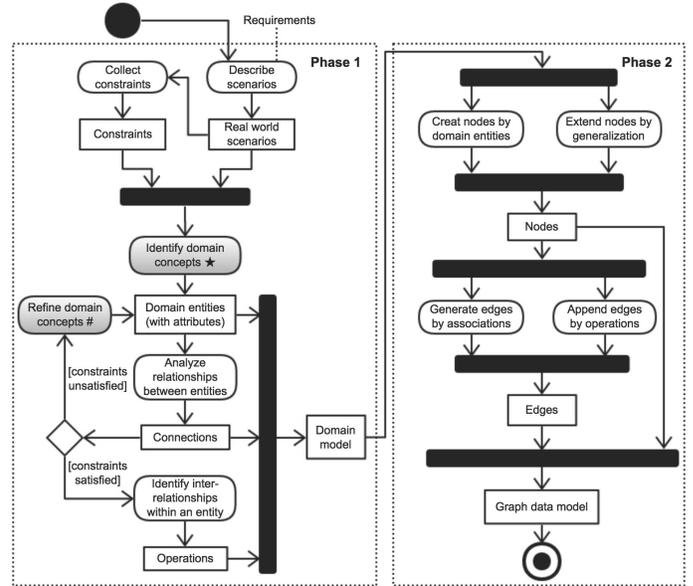


Figure 1. An overview of SuMo

2) *Invention*: Through invention, we create artifacts (new concepts or attributes) that are useful for the application domain. One of the cases is when dealing with the so-called “n-ary” relationships.

3) *Integration*: Remember, domain modeling is directed against graph database applications. Integration is used to merge those less important concepts into related ones.

4) *Split-off*: In contrast, some concepts, which are usually treated as attributes in general domain modeling, might be important “indexes” of that domain. According to scenarios and constraints, we will split off those important attributes to become new entities.

Besides, we add a specific activity in the adapted domain modeling: using operations to identify inter-relationships within a domain entity. These operations define the relationships between each object of a conceptual class, which is important to graph model since each node in graph is an object not a class. By ensuring the correctness of the domain model we are implicitly improving the graph data model.

B. Phase 2: Model transformation

In this phase, domain entities, attributes, connections and operations are mapped into different elements of graph model accordingly. We set up the following rules:

1) *Creating nodes by domain classes*: Each object of a domain class will be transformed into a node, attributes of this object will be properties of the corresponding node. Fig. 2 presents an example of creating nodes by domain classes.

2) *Extending nodes by generalization*: Generalization indicates that the subclass is considered to be a specialized form of its superclass. Corresponding node of subclass will be tagged with the type of superclass, the properties of this node will be extended accordingly. Fig. 2 presents an example of extending nodes by generalization.

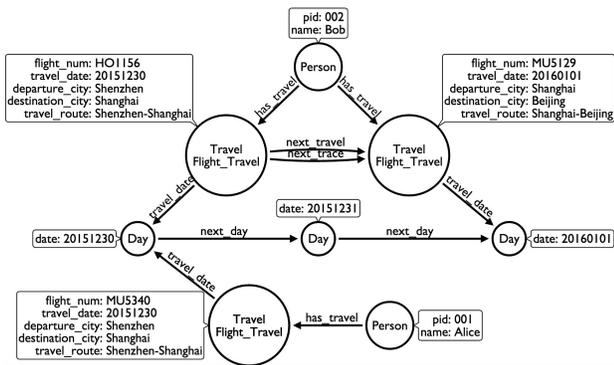


Figure 6. An instance of *S-Model*

For comparison, we build another graph model (*N-Model*) following a naïve approach (tuples are mapped to nodes and *Foreign Keys* are to edges) commonly used in mapping relational model to graph model.

V. EVALUATION AND RESULTS

We now present the experiments to evaluate SuMo in terms of query performance comparing with the naïve approach. All the experiments were executed in a machine with an Intel Core i5 processor, 2.8 GHz and 8 Gigabytes of RAM memory. We used Neo4j Community Edition 2.3.0 and set JVM heap to 4096K. All the queries are written in Cypher, a declarative graph query language provided by Neo4j. The datasets illustrated in Section IV were used to conduct our experiments, for the reason of confidentiality, part of the real data were allowed to be used (427,127 travel records).

Based on the scenario described in Section IV, we grouped 7 query sets including 4 simple query sets (set1-set4) and 3 rather complex query sets (set5-set7). Each set has 5 different queries that are homogeneous with respect to function and complexity. We chose input randomly, executed each query in all sets 10 times, measured the average execution time.

A. Simple queries

Fig. 7 presents the performance of simple queries (set1-set4). The *S-Model* performs consistently better than the *N-Model* for all of the queries, significantly outperforming in set3 and set4 that concern about `travel_route`. This is due to *S-Model* scans less subgraphs than the *N-Model* which spends more time in mapping airports or train stations to cities.

B. Complex queries

Considering complex queries (set5-set7), we found they can hardly be achieved only by Cypher in *N-Model* because these

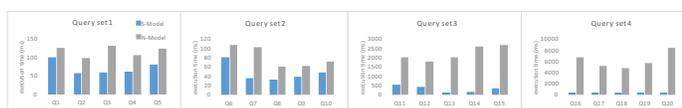


Figure 7. Performance of simple queries

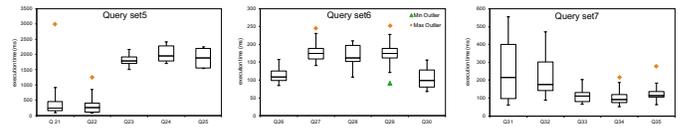


Figure 8. Performance of set5-set7 in *S-Model*

queries seek multi-step routes. Generally, we need some off-line processing to get them done in relational model. *N-Model* is directly mapped from relational model, the difficulty in dealing with these rather complex queries is similar. In *S-Model*, these queries can be accomplished in normal graph traversals. The performance of set5-set7 is shown in Fig. 8. The outliers fall in an acceptable range.

VI. CONCLUSION AND FUTURE WORK

In this paper, we presented an exploration of a systematic graph data modeling approach based on scenarios. Our preliminary approach contains an adapted modeling phase involving specific strategies for capturing domain concepts. Rules for converting domain model to graph model are defined in the phase of model transformation. The experimental results obtained are promising regarding SuMo’s potential to be used in real development. In future work, we will consider generating properties of edges in the model transformation phase. We also intend to refine the domain modeling phase by providing templates of scenarios.

REFERENCES

- [1] R. Angles, “A comparison of current graph database models,” in *Proceedings of the 2012 IEEE 28th International Conference on Data Engineering Workshops*. IEEE Computer Society, 2012, pp. 171–177.
- [2] C. Cattuto, M. Quaggiotto, A. Panisson, and A. Averbuch, “Time-varying social networks in a graph database: a neo4j use case,” in *First International Workshop on Graph Data Management Experiences and Systems*. ACM, 2013, p. 11.
- [3] A. Gonzalez-Beltran, E. Maguire, P. Georgiou, S.-A. Sansone, and P. Rocca-Serra, “Bio-graphiin: a graph-based, integrative and semantically-enabled repository for life science experimental data,” *EMBnet. journal*, vol. 19, no. B, pp. pp–46, 2013.
- [4] Y. Park, M. Shankar, B.-H. Park, and J. Ghosh, “Graph databases for large-scale healthcare systems: A framework for efficient data management and data services,” in *2014 IEEE 30th International Conference on Data Engineering Workshops (ICDEW)*. IEEE, 2014, pp. 12–19.
- [5] R. De Virgilio, A. Maccioni, and R. Torlone, “Model-driven design of graph databases,” *Conceptual Modeling*, pp. 172–185, 2014.
- [6] I. Robinson, J. Webber, and E. Eifrem, “Graph databases,” 2013.
- [7] “Neo4j GraphGist,” <https://github.com/neo4j-contrib/graphgist/wiki>, accessed: Nov. 21, 2015.
- [8] “Neo4j Top 7 Use Cases,” <http://neo4j.com/use-cases/>, accessed: Jan. 5, 2016.
- [9] R. De Virgilio, A. Maccioni, and R. Torlone, “Converting relational to graph databases,” in *First International Workshop on Graph Data Management Experiences and Systems*. ACM, 2013, p. 1.
- [10] D. W. Wardani and J. Kiing, “Semantic mapping relational to graph model,” in *2014 International Conference on Computer, Control, Informatics and Its Applications (IC3INA)*. IEEE, 2014, pp. 160–165.
- [11] D. W. Wardani and J. Kung, “Semantic mapping relational to a directed property hypergraph model,” in *2015 IEEE International Conference on Computer and Information Technology; Ubiquitous Computing and Communications; Dependable, Autonomic and Secure Computing; Pervasive Intelligence and Computing (CIT/IUCC/DASC/PICOM)*. IEEE, 2015, pp. 152–159.

An Agile Methodology for Reengineering Object-Oriented Software

Anam Sahoo, David Kung, and Sanika Gupta

Department of Computer Science and Engineering, The University of Texas at Arlington, USA

Abstract— Software maintenance is an important phase in the software development life cycle. More than 75% of maintenance efforts are enhancement. Currently, most enhancement projects are carried out in an ad hoc manner, depending on the knowledge and experience of the developers. Software reengineering aims to provide an engineering approach for software enhancements. In this paper, we present an agile reengineering methodology for object-oriented software. The methodology has a quick planning phase followed by a series of iterative reengineering phases. Each iteration consists of three legs: the reverse engineering leg, the reincarnation leg, and the validation leg. Academic and industry experiments show promising results.

Keywords and phrases: Software process and methodology, software maintenance, software reengineering, agile method, object-oriented software.

I. INTRODUCTION

Software maintenance typically consumes an average of 60% of software life costs, with enhancements being responsible for more than 75% of the costs [11]. These costs are a grand challenge for the current software community, in which tens of millions of lines of legacy code need to be modified during enhancement maintenance. The problem becomes even direr when the enhancement project is performed by engineers who do not have sufficient knowledge of the legacy system and documentation is inadequate or nonexistent. Software reengineering aims to provide an engineering approach for software enhancement. Current literature surveys reveal that there is a lack of a systematic reengineering methodology.

In this paper, we present a methodology for reengineering object-oriented software. It has three distinct phases: a release planning phase, iterative reengineering phase, and a system validation phase as shown in Fig. 1.

Each release begins with a quick agile planning phase, followed by an iterative reengineering phase consisting of a series of iterations. At the end of the release, an optional system validation phase is performed to validate the release before it is formally delivered to customers. The planning phase performs two activities. First, new requirements are identified and prioritized by applying information collection techniques and are based on a statement of work (SOW) from the customer. Second, new use cases and changes to existing use cases are derived. Finally, planning for release iterations is performed to produce a roadmap to guide the iterative reengineering activities. The iterative reengineering phase consists of a series of iterations. Each iteration has three legs: the reverse engineering leg, the reincarnation leg, and the validation leg as shown in Fig. 2. This is referred to as the N-process model.

The reverse engineering leg recovers design artifacts and helps to understand the existing system. It starts from a legacy code and has three major outputs: recovered design, recovered architecture, and recovered requirements in the form of use cases. There are techniques described in

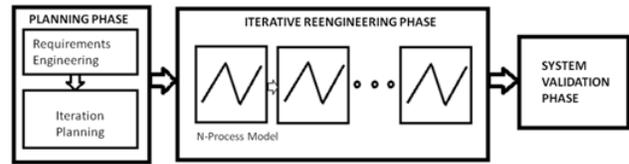


Figure 1. Agile OO SW reengineering methodology

[2,6,12,17-18,21,23,25,28,33,35] for recovering these design artifacts. The middle leg is the re-incarnation leg, which transforms the legacy system to a new working system. The third leg of the iterative reengineering phase focuses on validating the implementation against the intended design and requirements by preparing appropriate test cases. These include component level unit test cases, subsystem/system integration test cases, and system test cases. The system validation phase is meant to perform a formal release testing even though functional unit and integration testing have already been performed during the iterative reengineering phase. A formal system testing is conducted when a release candidate build is ready. Then a well tested release build is handed over to customer for customer acceptance testing.

The detailed step-by-step methodology for these phases is described in section II. Section III describes the application of the methodology to two academic experiments, which show significant improvements in project schedule and software quality. Section IV describes related work, followed by conclusions and future work in section V.

II. THE REENGINEERING METHODOLOGY

To illustrate the steps of the methodology, the Academic Advising Scheduler Web (AASW) system will be used as the legacy system. It is a software application written in Java and JSP with a MYSQL data base. It supports three types of users: administrators, advisors, and students. TABLE I lists the existing legacy requirements and use cases assumed to be already available to limit the scope of this paper. If missing, the needed use cases, can be recovered using techniques described in [6,43].

A. Planning Phase:

The planning phase performs two activities. First, new requirements are identified and prioritized by applying information collection techniques. Their impact on existing use cases is assessed, resulting in new, modified, and deleted

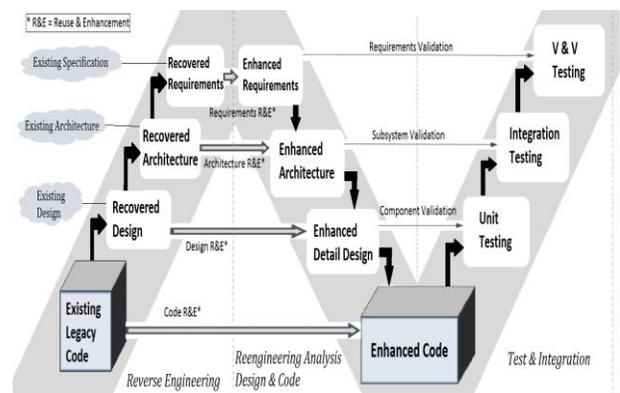


Figure 2. Each iteration of the agile reengineering phase

use cases, respectively. Second, planning for release iterations is performed to produce a roadmap to guide the iterative reengineering activities.

Identifying and Prioritizing Enhancement Requirements: Requirements are part of the contract of a software project. They specify the capabilities that the software system must deliver. Thus, correctly identifying and prioritizing enhancement requirements are critical to the success of a reengineering project. To identify and prioritize enhancement requirements, information collection techniques such as customer presentation, user survey, user interview, and literature survey are used. Next, new, modified and deleted use cases are derived. A new use case is derived if an application-specific verb-noun phrase is found or inferred from an enhancement requirement and the verb-noun phrase satisfies the following conditions: (1) it denotes a complete end-to-end business process of the application, (2) the business process begins with a user, (3) the business process ends with the user, and (4) the business process accomplishes a useful business task for the user. Sometimes, an enhancement requirement specifies that a piece of functionality needs be incorporated to an existing use case of the legacy system. Finally, existing use cases may no longer be needed due to the changing business environment. For AASW, the enhancement requirements identified during the planning phase are: R4: Users can change passwords. R5: Users must change system-generated temporary passwords when logs in for the first time. R6: Student ID and passwords must be encrypted before storing them in the database. From R4 above, we derive one verb-noun phrases: change password. This satisfies the four conditions for a use case described above. Therefore, a “Change password” new use case is derived. From both R5 and R6 above, we derive four modified use cases: “UC1: Create advisor,” “UC6: Create student,” “UC4: Login user,” and “UC7: Schedule appointment.” The existing use cases, UC1 and UC6 need modifications to incorporate encryption of temporary password. The UC4 needs modification to force the user to change the temporary password during login and encrypt the new password. The UC7 should be modified to include encryption of student ID while scheduling an appointment. Encryption of the password will be implemented in the new UC8: change password use case. TABLE II shows the impact to use cases. R4-R6 have priority 1, and must be completed in the first reengineering iteration.

The new, modified, and deleted use cases are assigned to iterations in this step. First, an agile estimation technique such as the poker game [7] is applied to obtain an effort

TABLE I. REQUIREMENTS AND USE CASES FOR THE EXISTING LEGACY SYSTEM

Legacy Requirements	Legacy Use Cases
R1: An Administrator can create, edit and delete advisors and define their privileges	UC1: Create advisor UC2: Edit advisor UC3: Delete advisor
R2: Advisors can login and specify their advising time.	UC4: Login user UC5: Update schedule
R3: Student can create account, login to it, and schedule an appointment with an advisor.	UC6: Create student UC4: Login user UC7: Schedule appointment

estimate for dealing with each of the use cases. An order to design, implement, delete, and test the use cases is derived, based on their dependencies and priorities. The use cases are then assigned to iterations according to the order.

B. Iterative Reengineering Phase:

The iterative reengineering phase consists of a series of iterations. Each iteration has three legs: the reverse engineering leg, the reincarnation leg, and the validation leg. It is worth noting a new term “reincarnation” here for the middle leg of the N-process model instead of using the over used term “reengineering” to avoid confusion. We interpret reengineering as a complete end-to-end methodology. Reverse engineering is performed only if design documentation that accurately represents the code does not exist. The reverse engineering leg starts from a legacy code and has three major outputs: recovered design, recovered architecture, and recovered requirements in the form of use cases. First, design artifacts such as class diagrams, sequence diagrams, and use cases can be recovered from existing code. High-level architectural design, domain model, and requirements for legacy system can then be derived from the recovered design artifacts. TABLE III summarizes all the needed artifacts and activities for each of new, modified, and deleted use cases.

In this paper we discuss only two reverse-engineered artifacts--- that is, implementation sequence diagram (ISD) and implementation class diagram (ICD). An ISD shows how the software objects interact with one another and in what order to process a user request. An ICD is an integrated view of the implemented classes, their attributes, methods, and relationships. Detailed steps for reverse engineering ISD and ICD are given below.

B.1. Reverse engineering ISD and ICD:

Step 1. Observe how a user uses the current system and describe the actor system interaction behavior. For example, a login use case behavior is as follows: (a) AASW displays any page having username and password fields in the page header area, (b) the user enters username and password and clicks “Submit,” (c) if the user authenticates correctly, AAWS displays the user’s dashboard, and (d) the user sees his dashboard.

Step 2. Identify nontrivial step(s) as follows: (a) If the step does not require background processing, or (b) if the system response simply displays a menu/input dialog, or (c) if the step displays the same system response for all actors, then it is a trivial step. but (d) if the system response is different for different actors, then it is a non-trivial step. In above example, the nontrivial step is (d).

TABLE II. ENHANCEMENT REQUIREMENTS AND IMPACT TO USE CASES

Enhancement Requirements	Use Cases	Category
R4: Users can change passwords.	UC8: Change password	New
R5: Users must change system-generated temporary passwords when logs in for the first time.	UC4:Login user UC8:Change password	Modified New
R6: Student ID and all passwords must be encrypted before storing them in the database.	UC1:Create advisor UC6:Create student UC4:Login user UC7:Schedule appointment UC8:Change password	Modified Modified Modified New

TABLE III. REENGINEERING ARTIFACTS AND ACTIVITIES FOR DIFFERENT USE CASE CATEGORY

Artifacts/Activities	Category of Use Case		
	<i>New</i>	<i>Modified</i>	<i>Deleted</i>
Reverse engineer ISD	Not needed.	Yes, needed, to be modified to take into account the enhancement requirements.	Yes, needed, to identify potential classes and methods to delete.
Reverse engineer ICD	Yes, needed, to identify classes and methods to reuse or extend	Yes, needed, to identify classes and methods to reuse or extend.	Yes, needed, to identify classes and methods to delete.
Construct expanded Use Case (EUC)	Yes, needed.	May be needed if actor-system interaction behavior need be changed.	No, not needed
Construct/modify Design Sequence Diagram (DSD)/ISD	New DSD - with new and existing classes from ICD. May apply software design patterns (SDP) such as adapter, controller, and facade.	Modified ISD, consider reusing existing classes from ICD.	Use ISD to identify classes and methods to delete.
Modify ICD	Add new classes and modify classes of ICD according to the DSD. Enhance design with SDP.	Add new classes and modify classes of ICD according to modified ISD. Enhance design with SDP such as adapter and facade.	Delete identified classes and methods from ICD.
Create New Test Cases	Needed for new classes and new methods, and classes affected by changes.	Needed for modified classes and methods, and classes affected by the changes.	No new test cases needed
Do Regression Test	Yes, needed.	Yes, needed.	Yes, needed.

Step 3. Identify the button that initiates the nontrivial step. Identify the action listener for the button. Trace the action listener handler code for objects and messages sent between them. Construct implementation sequence diagrams (ISD) to describe interactions between these objects. Techniques and tools for reverse engineering code to produce ISD and ICD are found in [13,17-18,21,23,25-26] and [13,18,22,26-28,30] respectively. Some of these tools could be used to reduce effort.

Reincarnation: This middle leg transforms the legacy system to a new working system. The new, modified and deleted use cases are already identified during planning phase. For each use case, depending on its category, the following three steps are performed: First, an individual use case is used to identify and recover implemented design and high-level architectural artifacts from the code. Second, the necessary additions, modifications, and deletions are made to those recovered artifacts. Finally, changes in the artifacts are incorporated into the existing code during the implementation phase.

Validation: The third leg of the iterative reengineering phase focuses on validating the implementation against the intended design and requirements by preparing appropriate test cases. These include component level unit test cases, subsystem/system integration test cases, and system test cases. The combined detailed steps for reincarnation and validation activities for new, modified, and delete use cases are as follows:

B.2.Reengineering for a New Use Case:

Treatment for new use cases is similar to forward engineering, except that existing the legacy code and some of the test cases may be reused. Forward engineering is described in various publications [1,3,14-16,18]. The steps are summarized as follows:

Step 1. For each new use case, describe how a user or actor will interact with the system to carry out the business process. This is called actor-system interaction modeling/design. Consider, for example, the “change password” use case as described in the planning phase A.1. It may go as follows: (a) the user select the “change password” tab after login, (b) the system asks the user to enter information: new pass word, and confirmation password, (c) user enter needed information and press enter,

(d) the system displays a confirmation message, and (e) the user sees the confirmation message that the new password is successfully stored in the system.

Step 2. Identify actor-system interaction steps that required background processing such as database access or user-dependent computation. For example, steps (c) and above require the system process user information and display the confirmation message to the user. We call such steps nontrivial steps.

Step 3. For each pair of nontrivial steps, produce a design sequence diagram (DSD) to describe how software objects would interact with each other through message passing (or function calls) to produce the output (e.g., the confirmation message) from the user input (e.g., new password and confirmation password). Software reengineering should reuse existing code as much as possible, classes of the ICD as recovered in section B.1. Extract classes from the DSDs along with their methods, attributes, and relationships such as call relationship and use relationship. Use these to modify the recovered ICD as follows. To facilitate identification of classes, methods, and relationships to be implemented, the changes are highlighted in the ICD accordingly. (a) If a class is not in the ICD, then add the class to the ICD along with all its methods and attributes extracted from the DSDs. (b) If a class is in the ICD but some of its methods/attributes extracted from the DSD are not in the ICD, then add these to the class in the ICD. (c) If a relationship is not in the ICD, then add the relationship to the ICD.

Step 4. Implement new classes and methods and modify the existing classes and methods as per the new DSD and modified ICD.

Step 5. Prepare and execute functional unit and integration regression test cases for new classes and methods, modified classes, and classes and methods affected by changes.

Step 6. Make the necessary code changes until the test cases successfully pass as per TDD.

Step 7. Maintain traceability to keep track of changes. Examples of DSD and ICD for “UC8: Change password” new use case for AASW code base A are shown in Fig. 3 and Fig. 4 respectively.

B.3.Reengineering for a Modified Use Case:

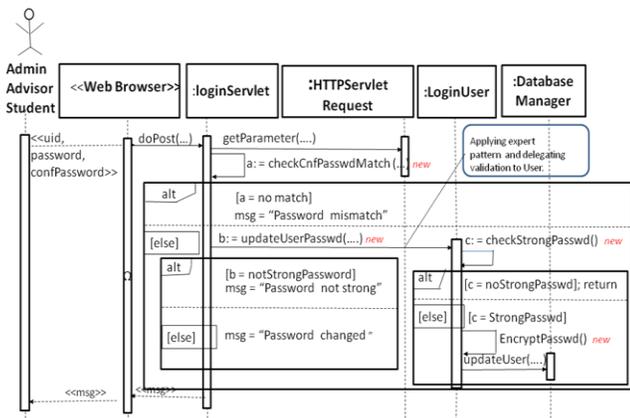


Figure 3. Change password DSD

Reengineering for modified use cases is the most common reengineering activity. The steps are as follows:

Step 1. Observe how a user uses the current system and reverse engineer the ISD and ICD as specified in the above reverse engineering section B.1.

Step 2. Modify the recovered ISD as required by the enhanced requirements. This step should attempt to reuse legacy code classes and is described in step 3 for new use cases in section B.2.

Step 3. Extract classes, methods and relationships from the modified recovered ISD and use them to modify the ICD, as described in step 4 for new use cases in section B.2.

Step 4. For each class method, prepare necessary unit test cases and subsystem/system integration regression test cases in parallel while incorporating the necessary code changes. Perform the code review, functional unit testing and integration regression testing immediately before and after any code modification is performed as per TDD.

Step 5. Maintain traceability to keep track of changes. Recovered/modified ISD for UC4 for code base A is shown in Fig. 5.

B.4.Reengineering for a Deleted Use Case:

Deleted use cases need to be handled carefully as follows: *Step 1.* Observe how a user uses the current system and reverse engineer the ISD as specified in the above reverse engineering section B.1.

Step 2. Identify the classes and methods of the ICD that need to be deleted.

Step 3. Identify and run necessary regression integration test cases for each deleted class and run before deleting any code. Check that functionality exists.

Step 4. Comment out classes and methods and make sure that no other modules are dependent on them. Comment out classes or methods that are not used.

Step 5. Run the system/subsystem integration regression test cases immediately after the code is commented out as per TDD. Make sure that the functionality is deleted. Identify and run selective impacted other regression test cases.

Step 6. Delete commented classes and methods and make necessary changes to the ICD.

Step 7. Maintain traceability to keep track of changes.

III. CASE STUDIES

In this section, we briefly present the impact of the methodology on schedule and quality of reengineering projects in comparison to doing it in any other way.

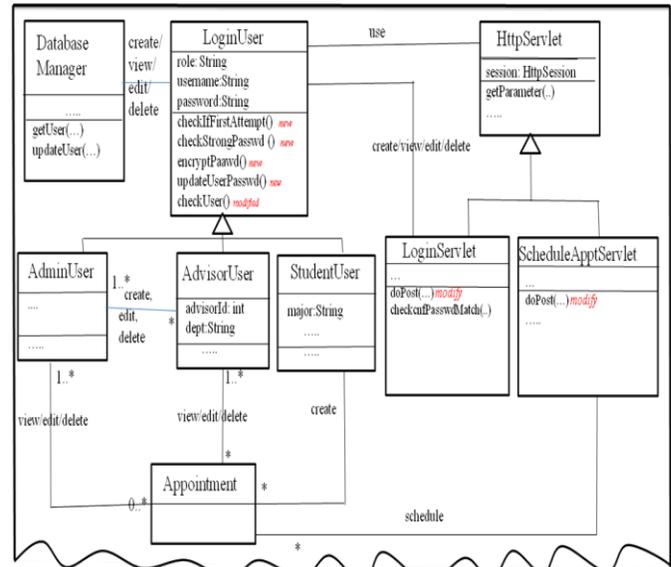


Figure 4. Recovered and modified ICD

Overview of Case studies: We conducted two academic case studies, as summarized in TABLE IV and an industry one. The academic case studies involved 30-31 graduate students in two different semesters using two legacy code bases of the AASW system. The third case study is for an industry project conducted by a local railroad company to reengineer a Driver-Assist legacy system to an Auto-pilot system involving a team of 8 experience engineers. For the first phase of the first case study, students divided into 7 teams were asked to do their reengineering assignments in their own ways. Then, the methodology was taught to the class. In the second phase, the code bases were swapped and asked to do the second reengineering assignments following the methodology. In the second case study, the eight teams were asked to learn and use the Rational Unified Process (RUP)[4] and Agile Unified Methodology (AUM)[16] for the first assignment. Then, teams used the methodology for the second assignment after swapping the code bases. For both the phases of case studies, data were collected from all the teams in the following two ways: (1) artifacts submission that include enhanced code and (2) the teams were asked to give demonstration of their working code to

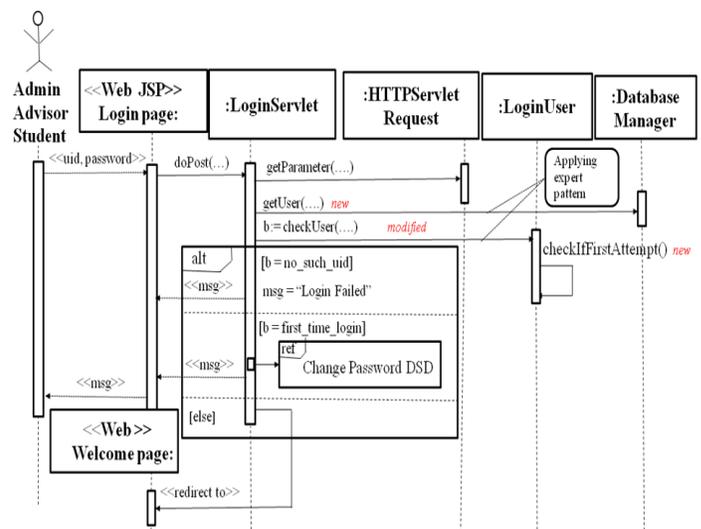


Figure 5. Login user recovered and modified ISD

TABLE IV. OVERVIEW OF CASE STUDIES

Case Studies		1		2	
# of Participants		30		31	
		CSE Students		CSE Students	
Teams		2-4	1, 5-7	1,3,5,7	2,4,6,8
AASW Project code base	Before learning the methodology	A	B	A	B
	After learning the methodology	B	A	B	A
	Duration	5 weeks	5 weeks	5 weeks	5 weeks

validate their claim. For the third case study, the railroad company used and shared their experience of using the methodology.

Analysis of the data collected from the first case study was performed by comparing different groups for the same legacy systems--that is, the quality of the enhanced code base A performed by the teams 2-4 in case study 1 using the methodology was compared with 3 out of the rest of the 4 teams (randomly picked) using ad hoc ways. But in the second academic case study, the analysis was performed comparing student performance with RUP and AUM with that of the methodology.

Analysis and Results Interpretation: Case study 1 results indicate that for both the code bases of AASW, following any ad hoc way, only one out of three teams submitted and demonstrated the working code. All the teams for both code bases just submitted the requirements as it was given to them without any prioritization, categorization, effort estimation, and traceability. In contrast, the teams who followed the methodology for both the code bases performed much better with 78-80% of enhancement requirements completed. All the teams submitted all the listed artifacts, which were evaluated to be of much better quality. The second case study results indicate that students following RUP or AUM could complete only 30 to 50% of needed enhancements. The team who submitted the working code just submitted some ISD and ICD but no other artifacts. In contrast, the teams performed much better with 75-80% enhancement completion as shown in TABLE V. All the teams submitted all the listed artifacts, which again were evaluated to be of

much better quality. Feedback from the industrial application indicated that the methodology significantly improved the schedule and quality of the company’s safety critical auto-pilot reengineering project in terms of requirements understanding, reverse engineering, enhancement of design, implementation, and testing. The estimated improvements was perceived to lead to a 25% reduction in schedule due to smooth integration and a 50% reduction in defects as compared to similar past reengineering projects.

There are a few limitations to our first two case studies: first, only two AASW legacy code bases were used to compare the effectiveness of the methodology. Second, even though students are inexperienced and the two projects have different code bases, the domain learning during the first assignment was definitely an advantage for the second assignment. Finally, the sample size is too limited to run any statistical analysis on. Despite all these limitations, these case studies show preliminarily that the participants performed much better using the proposed methodology compared to using RUP, AUM or doing it in any ad hoc way.

IV. RELATED WORK

There are very few studies performed on reengineering processes and methodology even though several plan-based and agile forward engineering processes and methodologies are currently used for reengineering activity. A framework-based agile reengineering process named PARFAIT using static structure of rational unified process (RUP) is described in [4], which explains how to rapidly provide an user with evolved versions of legacy system. [5] describes a segmentation reengineering process after recovering the analysis model from the procedural legacy C code and then partially transforming it to an object-oriented java code using design patterns. Another reengineering process for migrating legacy object-oriented systems to component based systems is described in [19], which suggests process metrics to improve code granularity and reusability. [32] explains an ontology based approach to reengineer legacy enterprise software to cloud computing environment. A reengineering process called “The Renaissance” is overviewed in [3]. It is a two-stage process for transforming legacy system to

TABLE V. PROJECT ARTIFACTS AND CODE COMPLETION STATUS COMPARISON

Code Base	Submitted Artifacts	Case Study 1							Case Study 2										
		Teams using any Ad-hoc way				Teams using The Methodology			Teams using process				Teams using the Methodology						
		2	3	4	Avg.	1	5	6	Avg.	1	3	2	4	Avg.	5	6	7	8	Avg.
Base - A	Teams →																		
	Requirements	x	x	x	100%	x	x	x	100%	x	x	x	x	100%	x	x	x	x	100%
	ISD/DSD	-	-	-	0%	x	x	x	100%	-	-	-	x	25%	x	x	x	x	100%
	ICD	-	-	x	33%	x	x	x	100%	-	-	x	-	25%	x	x	x	x	100%
	Test cases	-	-	-	0%	x	x	x	100%	-	-	-	-	0%	x	x	x	x	100%
	Traceability	-	-	-	0%	x	x	x	100%	-	-	-	-	0%	x	x	x	x	100%
	Running code	-	-	x	33%	x	x	x	100%	-	x	x	x	75%	x	x	x	x	100%
Completion %	0	0	10	3%	60	95	90	82%	0	40	30	35	24%	70	60	80	90	80%	
Base - B	Teams →	5	6	7	Avg.	2	3	4	Avg.	5	7	6	8	Avg.	1	2	3	4	Avg.
	Requirements	x	x	x	100%	x	x	x	100%	x	x	x	x	100%	x	x	x	x	100%
	ISD/DSD	-	-	-	0%	x	x	x	100%	-	-	x	x	50%	x	x	x	x	100%
	ICD	x	-	-	33%	x	x	x	100%	-	-	x	-	25%	x	x	x	x	100%
	Test cases	-	-	-	0%	x	x	x	100%	-	-	-	-	25%	x	x	x	x	100%
	Traceability	-	-	-	0%	x	x	x	100%	-	-	-	-	25%	x	x	x	x	100%
	Running code	x	-	-	0%	x	x	x	100%	-	x	x	x	25%	x	x	x	x	100%
Completion %	33	0	0	11%	65	95	75	78%	0	0	25	50	19%	60	70	90	80	75%	

x – Artifacts submitted and “-“ – Artifacts not submitted .

evolvable system: first, the strategic planning stage, and then, the continuous evolution stage.

Missing or outdated documentation in legacy projects is always an issue during reengineering. Many techniques, however, have been presented for reverse engineering artifacts, such as domain models[12], class diagrams[28], sequence diagrams [16,18,23,25,33], and use cases[7] from legacy code. An architecture recovery methodology using feature modeling is described in [24]. In this methodology, the top-down architectural element hypotheses are generated based on domain knowledge and verified using bottom-up tracing procedures. Finally, feature models are introduced bridging the gap between requirements and architecture. The rapidly changing business environment causes requirements to constantly change. However, missing legacy requirements or use cases is a common problem, and recovery is a very complex affair. A few use case recovery techniques are described in [6, 34]. The most important of all of these is the ability to trace all the reengineering activities from legacy code to requirements, to reincarnated design elements, and enhanced code. Tracing all around in reengineering using RETH tool is described in [10].

V. CONCLUSIONS AND FUTURE WORK

Reengineering is an important part of software maintenance in an industry in which the environment is constantly evolving and customer needs are ever-changing. This paper presents an agile methodology to reengineer object-oriented software, which focuses on a front end quick planning phase, an iterative development phase, and a system validation phase using test driven development approach. The application of the methodology on academic and industry experiments gives an early indication of improved code quality and project schedule over using RUP, AUM, or doing it in any ad-hoc way. The future work that remains to be completed seeks to extract a domain model and use cases from the recovered ICD and ISD iteratively. The agility of the methodology can also be improved by automating the manual steps and integrating them with existing reverse engineering tool sets. Finally, the code can be enhanced by restructuring with design patterns.

REFERENCES

- [1] M.R. Blaha & J.R. Rumbaugh, "Object-Oriented Modeling and Design with UML (2nd Edition)," Prentice Hall, 2004.
- [2] J. Borchers, "Invited Talk: Reengineering from a Practitioner's View -- A Personal Lesson's Learned Assessment," 15th European Conference on Software Maintenance and Reengineering (CSMR), 2011., pp. 1-2.
- [3] B. Bruegge & A.H. Dutoit, "Object-Oriented Software Engineering: Using UML, Patterns, and Java (3rd Edition)," Prentice Hall, 2009.
- [4] M.I. Cagnin & J.C. Maldonado, "PARFAIT: towards a framework-based agile reengineering process," Proceedings of the Agile Development Conference (ADC), 2003., pp. 22-31.
- [5] M.I. Cagnin, R. Pentead, R.T.V. Braga & P.C. Masiero, "Reengineering using design patterns," . Proceedings Seventh Working Conference on Reverse Engineering, 2000., pp. 118-127.
- [6] F. Chen, H. Zhou, H. Yang, M. Ward & W.C.C. Chu, "Requirements Recovery by Matching Domain Ontology and Program Ontology," IEEE 35th Annual Computer Software and Applications Conference (COMPSAC), 2011., pp. 602-607.
- [7] P. Claudia, M. Liliana & F. Liliana, "Recovering Use Case Diagrams from Object Oriented Code: An MDA-based Approach," Eighth International Conference on Information Technology: New Generations (ITNG), 2011., pp. 737-742.
- [8] M. Cohn, "Agile estimating and planning Prentice Hall Professional Technical Reference, 2006.
- [9] J.M. deBaud & S. Rugaber, "A software re-engineering method using domain models," International Conference on Software Maintenance, 1995., pp. 204-213.
- [10] G. Ebner & H. Kaindl, "Tracing all around in reengineering," IEEE Software, vol. 19, no. 3, 2002., pp. 70-77.
- [11] R.L. Glass, "Frequently forgotten fundamental facts about software engineering," IEEE Software, vol. 18, no. 3, 2001., pp. 112-111.
- [12] M. Hong, T. Xie & F. Yang, "JBOORET: an automated tool to recover OO design and source models," 25th Annual International Computer Software and Applications Conference (COMPSAC), 2001., pp. 71-76.
- [13] IBM tool, "Rational Rose Architectl," Available: <http://www.ibm.com/software/products/en/ratisoftarch>.
- [14] I. Jacobson, J. Rumbaugh & G. Booch, "Unified Software Development Process Addison-Wesley, 1999.
- [15] D.C. Kung, "On use case identification," . Boston, USA, Proc. of 25th Int'l Conf. on Software Engineering and Knowledge Engineering, June 26-29, 2013.
- [16] D.C. Kung, "Object-oriented software engineering: an agile unified methodology (International Student Edition)," McGraw-Hill, a business unit of the McGraw-Hill Companies, Inc, 2014.
- [17] Y. Labiche, B. Kolbah & H. Mehrfard, "Combining Static and Dynamic Analyses to Reverse-Engineer Scenario Diagrams," 29th IEEE International Conference on Software Maintenance (ICSM), 2013., pp. 130-139.
- [18] C. Larman, "Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design and Iterative Development (3rd Edition)," Prentice Hall, 2005.
- [19] E. Lee, B. Lee, W. Shin & C. Wu, "A reengineering process for migrating from an object-oriented legacy system to a component-based system," The 27th Annual International Conference on Computer Software and Applications (COMPSAC), 2003., pp. 336-341.
- [20] M. Lee & S. Park, "A methodology to extract objects from procedural software," The 24th Annual International Conference on Computer Software and Applications (COMPSAC) 2000., pp. 557-566.
- [21] L. Martinez, C. Pereira & L. Favre, "Recovering sequence diagrams from object-oriented code: An ADM approach," International Conference on Evaluation of Novel Approaches to Software Engineering (ENASE), 2014., pp. 1-8.
- [22] Object-Aid tool, "ObjectAid," Available: <http://objectaid.com/>.
- [23] T. Parsons, A. Mos, M. Trofin, T. Gschwind & J. Murphy, "Extracting Interactions in Component-Based Systems," IEEE Transactions on Software Engineering, vol. 34, no. 6, 2008., pp. 783-799.
- [24] I. Pashov & M. Riebisch, "Using feature modeling for program comprehension and software architecture recovery," The 11th IEEE International Conference and Workshop on Engineering of Computer-Based Systems, 2004., pp. 406-417.
- [25] A. Serebrenik, S. Roubtsov, E. Roubtsova & M. van den Brand, "Reverse Engineering Sequence Diagrams for Enterprise JavaBeans with Business Method Interceptors," The 16th Working Conference on Reverse Engineering, WCRE, 2009., pp. 269-273.
- [26] Sparxsystems tool, "Enterprise Architect," Available: <http://www.sparxsystems.com/>.
- [27] Tigris tool, "AgroUML," Available: <http://argouml.tigris.org> .
- [28] P. Tonella & A. Potrich, "Static and dynamic C++ code analysis for the recovery of the object diagram," International Conference on Software Maintenance, 2002., pp. 54-63.
- [29] M. Trudel, C.A. Furia, M. Nordio, B. Meyer & M. Oriol, "C to O-O Translation: Beyond the Easy Stuff," The 19th Working Conference on Reverse Engineering (WCRE), 2012., pp. 19-28.
- [30] UML-Lab tool, "UML-Lab," Available: <http://www.uml-lab.com/en/uml-lab/academic/>.
- [31] I. Warren & J. Ransom, "Renaissance: a method to support software system evolution," The 26th Annual International Computer Software and Applications Conference (COMPSAC), 2002., pp. 415-420.
- [32] H. Zhou, H. Yang & A. Hugill, "An Ontology-Based Approach to Reengineering Enterprise Software for Cloud Computing," IEEE 34th Annual Computer Software and Applications Conference (COMPSAC), 2010., pp. 383-388.
- [33] T. Ziadi, M.A.A. Da Silva, L.M. Hillah & M. Ziane, "A Fully Dynamic Approach to the Reverse Engineering of UML Sequence Diagrams," 16th IEEE International Conference on Engineering of Complex Computer Systems (ICECCS), 2011., pp. 107-116
- [34] Q. Li, S. Hu, P. Chen, L. Wu & W. Chen, "Discovering and Mining Use Case Model in Reverse Engineering," Fourth International Conference on Fuzzy Systems and Knowledge Discovery (FSKD), 2007., pp.431-436

A Content-Based Approach for Recommending UML Sequence Diagrams

Thaciana Cerqueira

Leandro Marinho

Franklin Ramalho

Department of Computer and Systems
Federal University of Campina Grande, Brazil

thaciana@copin.ufcg.edu.br

{lbmarinho, franklin}@dsc.ufcg.edu.br

Abstract

Software engineers usually have to face a large space of choices during the development process, including libraries/APIs, frameworks and UML models, which undermines their ability in finding the ones that best fit their needs. Recommender Systems appear as a solution to this problem since they have been applied successfully in other domains that suffer from similar issues. In this paper we propose to recommend UML Sequence Diagrams, a popular software artifact in many development processes, as an attempt to mitigate this problem. Our approach consists of: (i) a suitable representation of the users' information needs and sequence diagrams' content; and (ii) two content-based recommendation algorithms to recommend sequence diagrams that match the users' preferences. We performed a study with computer science subjects, where we generated recommendations with (ii) and measured the users' satisfaction upon these recommendations. Our preliminary results show that both algorithms are able to provide accurate recommendations.

1. Introduction

Created by OMG (Object Management Group) [1], the Unified Modeling Language (UML) describes a set of diagram types for describing different systems from various perspectives. Unfortunately, support for discovering and reusing diagrams is currently limited. From the many different types of media that can be retrieved through Web search engines (e.g. Google), software artifacts are not among them. For example, there is no easy way for a user express that she is searching for examples of real world UML Sequence Diagrams that use combined fragments and that asynchronous messages.

We propose to mitigate this kind of problem through recommender systems. Recommender Systems (RS) are great tools for filtering out and helping users to find relevant con-

tent. In some cases, they try to mimic the situation where the information needs of users are fulfilled by recommendations of like-minded or expert users [2]. Recommender Systems for Software Engineering (RSSEs), in particular, are software tools that can assist users in the activity of finding software artifacts. Moogole [3] is a search engine that uses metadata of software models for retrieving software artifacts. Our work is similar to this in the sense that we also exploit metadata information of software models (in this case UML Sequence Diagrams) to help users finding software artifacts of interest, but we do this by providing personalized recommendations based on the declared information needs of users.

RSSEs is a scarce area of research that has the potential to increase the quality and agility of software development [4]. While most of the related works in this area focus on source code recommendation (cf. Section 5), we investigate the recommendation of UML sequence diagrams, a yet unexplored problem in this field of research. This kind of recommendation has an important educational benefit for students and teachers since it allows one to find good examples of real world diagrams containing the features they want to learn/review.

Our approach consists in (i) a content-based recommender system's representation for sequence diagrams' features and (ii) the application of two classic content-based algorithms: a bag-of-words model borrowed from information retrieval [5], and another one based on our proposed representation [2]. We compare and evaluate both algorithms by means of a field experiment with computer science subjects well acquainted with UML diagrams and show that both approaches present reasonably accurate recommendations.

The remainder of this paper is organized as follows. Section 2 presents a brief background on RS and UML sequence diagrams. Section 3 introduces our approach. Section 4 presents the evaluation methodology and results. Section 5 presents related works and Section 6 concludes the paper with some final remarks.

2. Background

In this section, we present a brief summary of the main concepts related to this work: Recommender Systems, Bag of Words, and UML Sequence Diagrams.

2.1. Recommender Systems

RS are applications that aim to support users on their decision making process while interacting with large amounts of information. RS usually follow four main paradigms [6]: (i) **Collaborative-filtering**: The assumption is that users that shared similar interests in the past tend to share similar interests in the future. Collaborative-filtering algorithms usually rely on the historical data of users; (ii) **Content-based**: This type of recommendation is based on the content of the items being recommended; (iii) **Knowledge-based**: In this modality, the recommender algorithms exploit background knowledge about the recommendable items; (iv) **Hybrid**: Since each of the aforementioned approaches has advantages and disadvantages, a good combination of them would take advantage of the strengths and eliminate the weaknesses of each one.

2.2. Bag-of-Words

Many information retrieval systems represent queries and textual documents as a multiset of words [5]. This multiset, B , can be described as $B = \{(w_i, f(w_i)) | 1 \leq i \leq j\}$, where j is the amount of words of B , and f is a function that returns the number of occurrences for the word w_i in the current document. As an example, if we get a document composed by the sentence “sequence diagrams are UML diagrams“, we could represent B as

$$B = \{(sequence, 1), (diagrams, 2), (are, 1), (UML, 1)\}$$

This representation does not store the semantics, since there is only the number of occurrences for each word and the order it occurs does not matter for the model. For example, the text fragments *sequence diagrams are UML diagrams* and *UML diagrams are sequence diagrams* have very different semantic meanings, but have the same bag-of-words representation [7].

2.3. UML Sequence Diagram

The UML enables software developers to represent system models through different views. Each one of these views is modelled with abstractions called UML diagrams, which can represent structural and behavioural characteristics of the object-oriented paradigm. UML defines thirteen

diagrams [1], where each one is responsible for representing one or more features of the software being built. In order to represent the system behaviour, interaction diagrams allow the specification steps of interaction between objects and actors of the system. Below we describe some important elements that may compose a sequence diagram: (i) **Lifeline**: Represents system objects that participate in the exchange of messages. Figure 1 depicts two lifelines: *Web Customer* and *Bookshop*; (ii) **Message**: Represents the exchange of information between entities of an OO system. Figure 1 shows asynchronous and return messages. The message *!search inventory*, sent from *Web Customer* and received by *Bookshop*, is an asynchronous message where a line is drawn with a stick arrowhead; (iii) **Combined Fragments**: In its 2.0 version, combined fragments were added to sequence diagrams, *i.e.* new elements capable of changing the normal flow of execution. The combined fragments have 12 types (operators) and each one of them has a different meaning. Figure 1 shows the *loop* and *opt* combined fragments, where *opt* is nested with *loop*. While *opt* defines a fragment that must be executed only if the analysis of the expression on the guard returns a true value, the *loop* describes behaviours that need to execute the same sequence of commands iteratively; (iv) **Interaction Use**: Allows reusing other sequence diagrams. In Figure 1 the interaction use is represented by *ref*.

Figure 1 shows two objects that interact by means of messages to perform a purchase of books online. While the customer is buying, the interactive combined fragment *loop* is being executed. Within the interaction, the client may optionally view the description of the chosen book and purchase a book. When a customer completes a purchase, it must end the session according to the behavior specified in the sequence diagram checkout.

3. Our Approach

Our approach consists of: (i) suitable representations for the sequence diagrams and the users (Section 3.1); and (ii) algorithms that operate under the representation defined in (i) (Section 3.2). As a proof of concept, we compared two of such algorithms: Bag-of-Words (BoW) model borrowed from the information retrieval field and a Content-Based (CB) filtering algorithm. The reason for choosing recommendation algorithms that rely on the content of sequence diagrams instead of, *e.g.* collaborative filtering, is due to the fact that we did not find any publicly available repository of interaction data between users and sequence diagrams.

3.1. Sequence Diagram Features

Before generating recommendations, we need to choose a suitable representation for users' and items' profiles. To

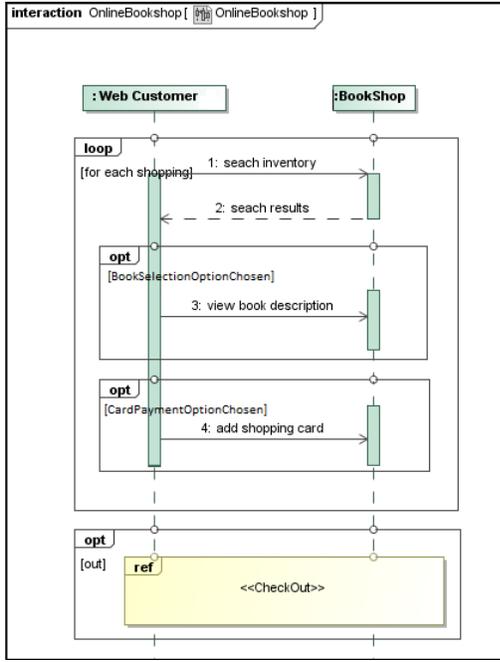


Figure 1: Example of a Sequence Diagram

this end, we have chosen to represent users and items as vectors in a space of features extracted from sequence diagrams. Below we describe the features we used for representing them. Follows the established representation: Presence of Lifelines (NL): We divided the number of lifelines into three groups, namely: small (from 1 to 3 lifelines), medium (from 4 to 6 lifelines) and large (over 6 lifelines) diagrams. Presence of Messages: Asynchronous (PAM); Return (PRM); Create (PCM); Delete (PDM); Presence of Combined Fragment: Conditional (PCoCF); Iterative (PItCF); Break (PBCF); Concurrent (PCuCF); Weak (PWCF); Strict (PSCF); Negation (PNCF); Critical (PCrCF); Ignore (PIgCF); Consider (PCsCF); Assertion (PACF); Presence of Interaction Use (PIU); Presence of Actors (PAc); Presence of State Invariant (PSI).

In the CB approach the user and item profiles are defined by binary vectors, as described above, of the form

$$\vec{p} = (\text{NL}, \text{PAM}, \text{PRM}, \dots, \text{PACF}, \text{PIU}, \text{PAc}, \text{PSI})$$

In the BoW model, users and items are represented as vectors of strings extracted from the sequence diagrams XMI file. Each string represents a sequence diagram feature, e.g. “uml:Lifeline” and “uml:Actor”, as can be seen in column “BoW” shown in Table 1. Each string is weighted with the well known tf-idf (term frequency - inverse document frequency) scheme where the string weights are directly proportional to their frequency in the sequence diagram and inversely proportional to their appearance over all sequence diagrams in the repository. In information re-

trieval terms, the user is the query and the sequence diagrams are the documents.

3.2. Recommender Algorithm

CB algorithm: A RS can be formally described as a function where U and I are the set of users and items (sequence diagrams in our case), and s is a function that estimates the utility of $i \in I$ to $u \in U$.

$$s : U \times I \rightarrow \mathbb{R} \quad (1)$$

The recommendation list is computed in two steps. First the similarities between the target user and the item profiles are calculated. Next, the n nearest sequence diagrams are recommended (aka top- n recommendation) to the user, according to Section 3.1. For the similarity computation we have used the well known and widely used cosine similarity measure. The cosine similarity receives two vectors as input and returns 1 if they have maximum similarity and 0 otherwise. More formally, for two m -dimensional profile vectors \vec{x} and \vec{y} the cosine similarity is computed by

$$\text{sim}(\vec{x}, \vec{y}) = \frac{\sum_{i=1}^m x_i \cdot y_i}{\sqrt{\sum_{i=1}^m x_i^2} \cdot \sqrt{\sum_{i=1}^m y_i^2}} \quad (2)$$

where x_i and y_i are the i -th component of vectors \vec{x} and \vec{y} respectively. Now, the top- n items for the target user $u \in U$ are computed as follows:

$$\text{top-}n(u) := \underset{i \in I}{\text{argmax}}^n \text{sim}(\vec{u}, \vec{i}) \quad (3)$$

BoW algorithm: A BoW can be formally described as a set of documents $C = \{D_1, \dots, D_l\}$, where l is the amount of documents, and each document $D_p \in C$ is represented as a multiset $D_p = \{(w_{p_q}, f(w_{p_q})) | 1 \leq q \leq m\}$, where m is the amount of words of D_p and $f(w_{p_q})$ is a function that returns the number of occurrences for the word w_{p_q} . Each word w_{p_q} is extracted from the text file representation for D_p . Considering the diagrams at XMI format as text files (since the format is XML-based), base in an information retrieval algorithm [7].

Each user Y_r is represented as a set $Y_r = \{t_1, \dots, t_w\}$ of search terms. For each user, the algorithm calculates an score (Equation 4) for every document $D_p \in C$.

$$\text{score}(Y_r, D_p) = \sum_{s \in Y_r} \text{tf-idf}_{s, D_p} \quad (4)$$

Lastly, the top- n items for the user $Y_r \in V$ are computed as follows (Equation 5):

$$\text{top-}n(Y_r) := \underset{z \in C}{\text{argmax}}^n \text{score}(Y_r, z) \quad (5)$$

4. Evaluation

In this section we present the experimental protocol used, results and threats to validity. In order to establish a proof of concept to the proposed profiles and also discover which algorithm makes the best recommendations, we have performed an experiment for comparing the proposed UML Sequence Diagrams recommenders.

4.1. Planning:

For evaluating the recommendations, the user and item profiles are created according to the preferences indicated by participants and the diagrams parsed by the system, respectively. Next, for each recommendation algorithm, the recommendations are generated and displayed to the participants that in turn must accept or not the recommended sequence diagrams displayed in a top-5 list. Finally, before the end of the experiment, the participants must judge the experiment as good suggestions or not, using the following levels of satisfaction: {Very Satisfied, Satisfied, Indifferent, Dissatisfied, Very Dissatisfied} (aka likert scale). [8].

4.2. Collecting and Formatting Data:

Given that we did not find any publicly available repository of sequence diagrams, we decided to construct our own experimental database. For that, we used the *Magic Draw* tool¹ to reverse engineer source code from the *FindBugs*² project. This tool also allowed us to recover and parse the XMI file generated from sequence diagrams. By parsing the XMI file, we identified all elements composing the sequence diagram. From this process, we were able to build 24 diagrams. Additionally, we generated 20 diagrams manually. These diagrams were generated because some features (e.g. PBCF, PWCF) were missing in the diagrams generated by Magic Draw. They were based on examples of literature, totaling 20 examples. For formatting the user profiles, each subject from the experiment must answer the questionnaire described in Table 1, column "Are you interested in viewing...". Each question relates to a specific features vector and the user should inform which features are of his/her interest. After constructing the user profiles, we calculated the cosine similarity between the target user and the item profiles vector and computed top-5 recommendation lists for BoW and CB approach.

Table 1 contains some examples about the user profiles.

¹<http://www.nomagic.com/products/magicdraw.html>

²<http://findbugs.sourceforge.net>

Table 1: The example form of interest: Content-based mapping (CB), Bag-of-words mapping (BoW)

Id	Are you interested in...	CB	BoW
1	diagrams of which size? (Small or Medium or Large)	NL	"uml:Lifeline"
	messages...		
2	↔creation message? (Yes or No)	PCM	"uml:Message", "message-Sort=createMessage"
	combined fragment...		
3	↔combined fragments of type assertion?? (Yes or No)	PACF	"uml:CombinedFragment", "interactionOperator=assert"
	sequence diagrams with...		
4	↔interactions of use ("req")? (Yes or No)	PIU	"uml:InteractionUse"
5	↔actors ("actor")? (Yes or No)	PAC	"uml:Actor"
6	↔invariant state? (Yes or No)	PSI	"uml:StateInvariant"

4.3. Selection of Subjects:

The experiments were conducted with volunteer students the software design discipline in the computer science course from the Federal University of Campina Grande. The experiment included 26 participants. The approach was presented to the group in a workshop where instructions for using the tool were explained.

4.4. Experimental Design:

We had an unpaired comparative experiment where, for each participant, we randomly selected the recommendation algorithm, being transparent to the user which method was used. Thus, half of the subjects received recommendations from the BoW model and the other half from the CB approach. The effectiveness of the system is related to its ability to perform good recommendations. Thus, the central question of this research was to investigate the accuracy and the level of satisfaction of the users considering the recommendations of the two approaches investigated. As evaluation metrics we used *precision* (Equation 6) which is a well known and widely adopted metric in the information retrieval and recommender systems literature [9]. The precision for a given user $u \in U$ is defined as follows:

$$precision = \frac{|\{relevant\ items\} \cap \{recov.\ items\}|}{|\{recov.\ items\}|} \quad (6)$$

4.5. Questions and Hypothesis Formulation:

Our experiment addresses the research questions, using the null hypotheses respectively. RQ1: Given the same user profile, does the content-based and bag-of-words differ in

precision? and H_{1-0} : The precision of the two approaches is equal. RQ2: From the point of view of user satisfaction, which of the two approaches is better? and H_{2-0} : The satisfaction for both approaches is the same.

4.6. Results

We used the *Wilcoxon-test* on the precision values computed for each subject, which reached a *p-value* of 0.3543, indicating that there is no statistical difference between the CB and BoW approaches. Thus, it was not possible to refute H_{1-0} presented in Section 4.5. As a reinforcement, we calculated the *Cohen d* value. Cohen [10] suggests a scale able to identify the impact of the intended effect that, in this experiment, is the difference of the precision between the algorithms. On this scale, a value of up to 0.2 is considered of small effect, from this one up to 0.5 is a medium effect, 0.8 and above represents a large effect. Our results for this calculation is 0.4303, which represents almost no effect.

From the evaluation results, the answer to RQ1: *Given the same user profile, the content-based and bag-of-words differ in precision?* No. Figure 2 shows the boxplot of the precision for both approaches. The CB approach has the highest precision in terms of the median, and this result may provide a positive evidence of the relevance of this algorithm, that will be further analyzed.

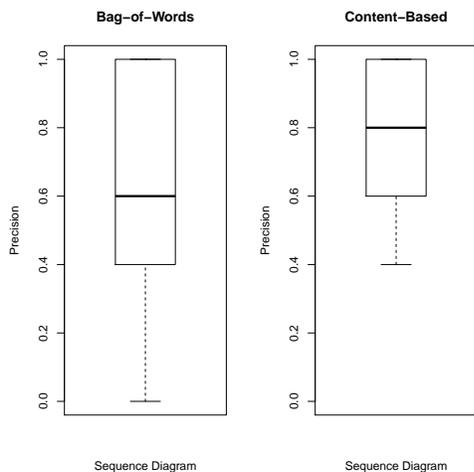


Figure 2: Boxplot - Comparison of Approaches

From a total of 44 sequence diagrams from the database, the experiment recommended 24 sequence diagrams from the total of 26 searches. Figure 3 shows the distribution of responses by recommended and accepted, summarizing the number of recommendations and sequence diagrams accepted for each of the approaches analyzed. This result

demonstrates that the acceptance of the approaches is relevant and that users accepted more sequence diagrams recommended by the CB algorithm.

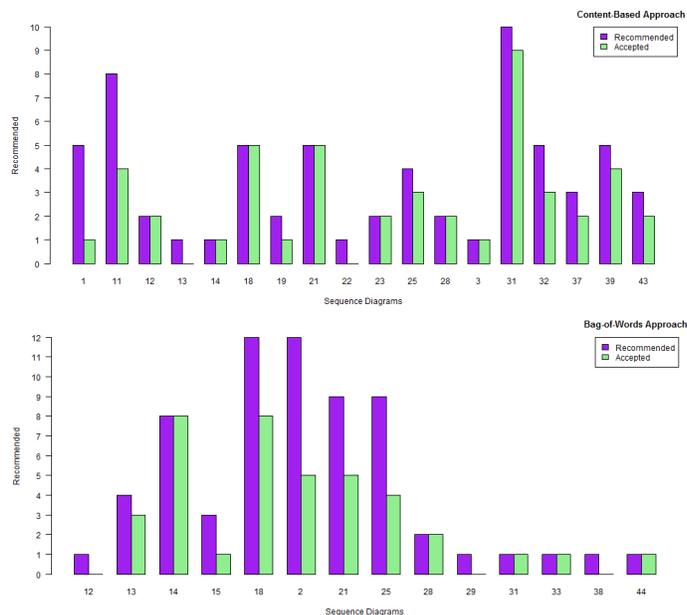


Figure 3: Recommendations Accepted by Approaches

Considering the sequence diagram characteristics displayed in Table 1, we can see that some characteristics were searched, selected and recommended more often than others by the users, such as asynchronous and return, in addition to combined fragment of the conditional (*opt* and *alt*) and iterative (*loop*) type. The most frequently chosen characteristics are also the most frequently recommended ones by the approaches, as we can see in Figure 4. One possible interpretation of this result is that both approaches are recommending sequence diagrams that match the characteristics requested by the users, but also that the subjects had a bias towards features they are already familiar with.

Regarding the answer to RQ2: *From the point of view of user satisfaction, which of the two approaches is better?* Two approaches achieve equal performance. The *Kruskal-Wallis chi-squared* indicates that there is a no statistical difference between the CB and BoW as concerning satisfaction, because *p-value* of 0.2987 and *Vargha-Delaney A measure* return *A measure* exactly 0.5.

4.7. Threats to Validity

In this section we describe the main threats to the validity of this work. **Internal Validity:** We had a low number of

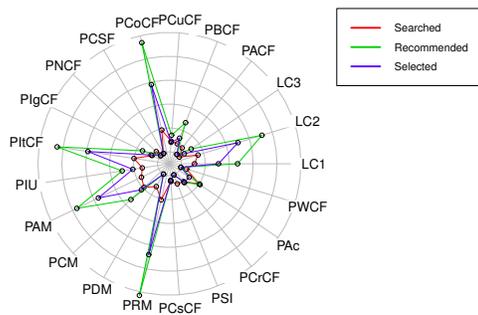


Figure 4: Item Profile Characteristics

sequence diagrams, which may result in cases where there is no sequence diagram covering the features of interest of the users. To address this threat, we manually created sequence diagrams to increase the number of characteristics available. **External Validity:** We have a low number of participants, which may lead to problems related to the statistical significance of the results.

5. Related Work

Most works aim to increase the reusability, providing ease of maintenance, improving productivity and making suggestions according to the preferences of the developer. Cubranic et al. [11] exploits recommender systems for bug fixes. Ye et al. [12], Lozano et al. [13], and McCarey [14] propose to recommend classes and methods based on the current class being used by the developer. Other works go beyond recommendation methods and indicate artifacts based on the bug fix process or recommend design patterns. The aforementioned works are important and represent an emerging area where information retrieval and recommender system techniques are used for searching and recommending software engineering artifacts. This paper complements these works by being one of the first research efforts (according to the reviewed literature) on using sequence diagrams recommendations.

6. Conclusion and Future Work

In this paper we proposed content-based recommender systems for recommending sequence diagrams. We have compared two recommendation models: a bag-of-words model borrowed from the information retrieval field and a classic CB algorithm. We conducted an experiment where we did not find statistical difference between the approaches considered. However, users accepted more sequence diagrams recommended by the CB algorithm. It is impor-

tant to emphasize that when filling the user profile by the users the most frequently chosen characteristics are also the most frequently features present in the diagrams recommended by the approaches. Hence, our study can serve as basis for future works aimed at identifying the main behavioral features of interest to developers when specifying UML design. As an ongoing work, we are currently investigating several other recommendation algorithms to be incorporated in our approach. In order to address current threats and reach more precise results, we intend to design and perform a new experiment with a larger database of user profiles and sequence diagrams.

7. Acknowledgments

This work was partially funded by the EU-BR BigSea project (MCTI/RNP 3rd Coordinated Call).

References

- [1] OMG. Introduction to OMG's unified modeling language (UML). [Online]. Available: http://www.omg.org/gettingstarted/what_is_uml.htm
- [2] X. Amatriain, A. Jaimes, N. Oliver, J. M. Pujol, F. Ricci, L. Rokach, and B. Shapira, *Recommender Systems Handbook*, F. Ricci, L. Rokach, B. Shapira, and P. B. Kantor, Eds. Springer US, 2011.
- [3] D. Lucrédio, R. de M. Fortes, and J. Whittle, "Moogle: a metamodel-based model search engine," *Software & Systems Modeling*, vol. 11, no. 2, pp. 183–208, 2012.
- [4] M. Robillard, R. Walker, and T. Zimmermann, "Recommendation systems for software engineering," *Software, IEEE*, vol. 27, no. 4, pp. 80–86, July 2010.
- [5] D. Metzler, "Beyond bags of words: Effectively modeling dependence and features in information retrieval," *SIGIR Forum*, vol. 42, no. 1, Jun. 2008.
- [6] D. Jannach, M. Zanker, A. Felfernig, and G. Friedrich, *Recommender Systems: An Introduction*. Cambridge University Press, 2010.
- [7] C. D. Manning, P. Raghavan, and H. Schütze, *Introduction to Information Retrieval*, 2008.
- [8] Q. Li, "A novel likert scale based on fuzzy sets theory," *Expert Syst. Appl.*, vol. 40, no. 5, pp. 1609–1618, Apr. 2013.
- [9] P. Resnick and H. R. Varian, "Recommender systems," *Commun. ACM*, vol. 40, no. 3, pp. 56–58, Mar. 1997.
- [10] S. Piasta and L. Justice, *Cohen's d statistic*. Thousand Oaks, CA: SAGE Publications, Inc., vol. In N. Salkind (Ed.), *Encyclopedia of research design*.
- [11] D. Cubranic, G. Murphy, J. Singer, and K. Booth, "Hipikat: a project memory for software development," *Software Engineering, IEEE Transactions on*, no. 6, pp. 446–465, June 2005.
- [12] Y. Ye and G. Fischer, "Reuse-conducive development environments," *Automated Software Eng.*, vol. 12, no. 2, pp. 199–235, Apr. 2005.
- [13] A. Lozano, A. Kellens, and K. Mens, "Mendel: Source code recommendation based on a genetic metaphor," in *ASE, 2011 26th IEEE/ACM International Conference on*, Nov 2011, pp. 384–387.
- [14] F. McCarey, "Agile software reuse recommender," in *ICSE 2005. Proceedings. 27th International Conference on*, May 2005, pp. 652–.

Software Clustering using Hybrid Multi-Objective Black Hole Algorithm

Kawal Jeet

Department of Computer Science
D.A.V. College,
Jalandhar, India
kawaljeet80@gmail.com

Renu Dhir

Department of Computer Science and Engineering
Dr. B. R. Ambedkar N. I. T.,
Jalandhar, India
dhirr@nitj.acin

Abstract—Software clustering is the process of organizing software units into appropriate clusters so as to efficiently modularize complex program structure. In this paper, we investigate the use of hybrids of Black Hole algorithm (developed using weighted aggregation, auxiliary archive and Genetic Algorithm) to optimize multiple objectives for clustering of android mobile applications. It is empirically and statistically observed that multi-objective Black Hole algorithm when improved using Genetic Algorithm and auxiliary archive outperforms Two-Archive algorithm and its counterparts.

Keywords- *bio-inspired algorithm, edgessim, nature-inspired algorithm, serach based software engineering, software clsutering*

I. INTRODUCTION

Human beings are always inspired by nature. Over the past couple of decades, a large number of complex research problems have found their solutions in nature-inspired algorithms such as Black Hole (BH) algorithm, Genetic Algorithm (GA) etc. BH algorithm [1] is inspired by the black hole theory of the universe and GA is inspired by Darwin's survival of the fittest. Literature has a many instances where nature-inspired algorithms are applied to various fields of software engineering such as software testing [2], software effort estimation [3], and software clustering [4-7] etc. Software clustering refers to the placement of software units in an appropriate cluster which is useful to identify the cluster responsible for a particular functionality. It not only improves the structure of the system but also enhances the system comprehension. It is hence useful in both the development and maintenance of a software system [8].

Large numbers of companies are developing mobile applications for the users of their domain. The developers in these companies are in immense stress to produce high-quality applications within deadlines. So, need to develop automated techniques to improve their maintainability have been aroused. It is believed that well-clustered mobile applications are easy to maintain. In this paper, BH algorithm along with its hybrids is applied for modularization of five android applications (described in Table I). The prime contributions of this research work are listed below.

- Formulation and investigation of the use of BH algorithm as multi-objective optimization technique for the process of software modularization of android mobile applications.
- Investigation of the impact of hybridizing BH algorithm with GA and auxiliary archive.

- Comparison of modularization results of proposed hybrid approaches to that of existing Two-Archive approach [7].

TABLE I. DESCRIPTION OF THE TEST PROBLEMS

Software System	Modules in MDG	Edges in MDG	System Description
Foursquare	54	6	Open source popular game
Sudoku	74	5	Open source popular game
Apps Organizer	135	13	Open source Organizer
Punjab Kesari	1234	32	Proprietary famous Punjabi newspaper (Developed by 'Converse New Media')
Desi coupon	244	4	Proprietary advertisement management app (Developed by 'Iniz Solutions' & yet to be launched)

II. LITERATURE REVIEW

Various search based optimization techniques have been applied to software clustering in past. A remarkable work in this field includes the use of GA and Hill Climbing algorithm (Bunch tool) [6] for automatically clustering software. They used the representation of the given software as a Module Dependency Graph (MDG) and Modularization Quality (MQ) is optimized to get desired clustering efficiently. MQ is further defined as the ratio of cohesion and coupling [6]. Praditwong et al. [7] [9] used six objectives for automatic software clustering and this approach outperforms Bunch tool. The authors of [10] used multi-objective GA for software modularization. In another work [11], the sum of intra-edges, inter-edges and the number of changes between original and updated clustering are used as fitness objectives using NSGA-II. This technique has been found to be successful for re-clustering. In another work [12], the authors used cooperative clustering for software modularization on the basis of MQ. With increase in size of problem, performance of this approach degrades. Particle Swarm Optimization (PSO) [4] and BH [13] algorithm has also been used for software clustering using MQ as optimization objective. Mkaouer et al. [14] applied NSGA-III algorithm for modularization of software using seven objectives. The approach is applicable if evolutions of the software are carefully maintained.

III. SOFTWARE MODULARIZATION USING BLACK HOLE ALGORITHM

BH algorithm is an optimization algorithm that searches for optimal solution on the basis of a set of objectives (mentioned

in Table II) often conflicting with each other. The general structure of BH algorithm is shown in Figure 1. To implement BH, the population of individuals is initialized using (1).

$$x_i^j = 1 + rand(0,1)(n - 1) \quad (1)$$

where $i=1,2,\dots,Pop$ (Pop is the population size as described in Table III); $j=1,2,\dots,n$ (n is the number of modules to be clustered). The control parameters to be used for implementing Black Hole algorithms are shown in Tables III. Since the BH algorithm is random, so each experiment has been conducted repeatedly 30 times, and the results thus obtained are analyzed and compared to that of existing Two-Archive algorithm based approach [7, 9]. NP, NAE, NIE, NCP, NCD and NIP described in Table II have been used as metrics for comparison. The problem of software clustering is formulated as a minimization problem. NAE (Table II) is a maximization objective and is reformulated as minimization objectives by negating its value.

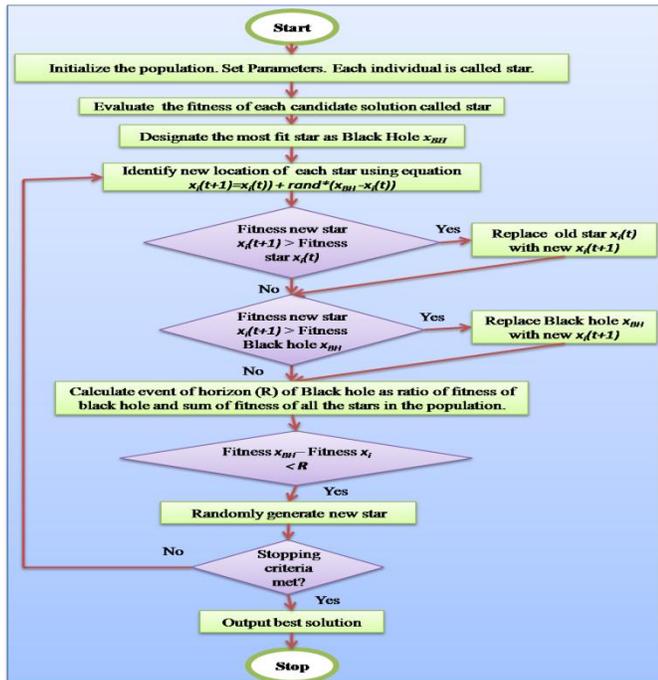


Figure 1. Black Hole Algorithm

A. Multi-Objective Weighted Black Hole Algorithm (MOWBH)

MOWBH algorithm is applied to the problem of software clustering. In order to calculate the fitness of an individual, weighted sum approach has been used. In this approach, all g objectives (f_k) mentioned in Table II are combined to make a single objective (F) as shown in (2). Use of random weights leads to sufficient diversity to obtain good quality clusters. The sum of weights of all the g objectives should be 1. This approach is easy to implement and widely used for multi-objective optimization. To overcome negative impact of randomness, the algorithm is executed 30 times with different random weights and the solution with least value of F is selected as the output [15].

$$F = \sum_{k=1}^g w_k f_k \text{ and } \sum_{k=1}^g w_k = 1 \quad (2)$$

These algorithms are highly dependent on the weights and in case of conflicting objectives; allocation of weights is

sometimes difficult. To overcome these problems, we investigated the use of Pareto optimization approaches [13] for optimizing the modularization of mobile applications.

TABLE II. OBJECTIVES TO BE OPTIMIZED

Objective	Optimization	Description
Number of Clusters (NP)	Minimize	Lesser the number of clusters more is the number of modules per cluster [7]
Number of Intra-edges (NAE)	Maximize	Dependencies among modules in the same cluster [7]
Number of Inter-edges (NIE)	Minimize	Dependencies among modules in different clusters [7]
Number of Modules per Cluster (NCP)	Minimize	It is conflicting to objective NP. It tends to create equal sized clusters [7]
Number of Cyclic Dependencies (NCD)	Minimize	Dependencies such that modules in cluster A depends on modules in cluster B and some other modules of cluster B depends on modules in cluster A [11]
Number of Isolated Clusters (NIP)	Minimize	Clusters with a single module [9]

TABLE III. CONTROL PARAMETERS FOR SOFTWARE CLUSTERING

Parameter	Value	Comments
Population size (N_c)	200	Manually tested by repeated executions of the algorithms.
Generations	10 * n or When output does not change for 300 consecutive generations.	Stopping criteria
Number of variables to be optimized (n)	Number of modules to be decomposed.	Each individual is composed of n decision variables.
Size of REP	1% of the size of population	To keep track of best (non-dominating) solutions
Crossover function (for GA)	Arithmetic	$Child = R1 * Parent1 + R2 * Parent2$ Where $R1, R2$ are independent random numbers between 0 and 1. Ideal value for software Clustering (found by manual testing): 0.6.
Mutation function (for GA)	Uniform	Ideal value for software Clustering (found by manual testing): 0.02.

B. Multi-Objective Hybrid Black Hole Algorithm (MOBH)

In this work, an auxiliary archive has been used to store Pareto front. Hyper-cubes have been used to maintain the best solutions for each iteration of the algorithm [16]. Although, the algorithm is very efficient in identifying optimal solutions but as the size of the problem increases, these algorithms tend to get stuck at local optima and the outputs are hence not globally optimal. In order to recover these algorithms from local optima, GA is used [5]. It leads to develop hybrid for MOWBH and MOBH called MOWBHGA and MOBHGGA respectively. The algorithms thus developed are shown in Figure 2 and 3 respectively.

IV. EXPERIMENTAL RESULTS AND ANALYSIS

In order to validate the clustering process, MoJoFM and EdgeSim have been used as assessment criteria. MOWBHGA and MOBHGGA are used for clustering of sample mobile applications (Table I) and results are compared to existing Two-Archive approach [7].

1) *MoJoFM as Assessment Criteria*

Let A be the automatic clustering and B be the reference cluster structure of an object-oriented system developed [17].

$$MoJoFM = \left(1 - \frac{mno(A, B)}{\max(mno(\forall A, B))} \right) * 100$$

$mno(A, B) = \min(\text{move and join operations to transform A to B})$,
 $\max(mno(\forall A, B)) = \text{most distant decomposition from reference decomposition}$.

2) *EdgeSim as Assessment Criteria*

$$EdgeSim = \frac{Weight(\gamma)}{Weight(E)} * 100$$

Where E is the set of all edges in a given MDG and γ is the set of inter-edges (inter-edges in A are inter-edges in B) or intra-edges (intra-edges in A are intra-edges in B).

Higher the value of MoJoFM and EdgeSim, better is the clustering.

Step 1 [Initialize population]:

Step 1.1: Encode and initialize the population of candidate clustering solutions. Each individual is called a *star*. Set parameters as shown in Tables III.

Step 1.2: Evaluate the fitness of each candidate in the population on the basis of combined weighted objective F calculated by using (2).

Step 1.3: Designate the solution with the least value of F as Black Hole (x_{BH}).

Repeat Steps 2-5 until stopping criteria is met (as indicated in Table III).

Step 2 [Identify new possible solutions]: For each iteration t , identify new location of star ($x_i(t+1)$) for each i^{th} clustering ($x_i(t)$)

$$x_i(t+1) = x_i(t) + rand * (x_{BH} - x_i(t))$$

Step 3 [Search for a better solution]: Evaluate fitness of each new clustering $x_i(t+1)$. If new candidate is better than the current candidate, then replace the current solution with this new. This is required to locally search for a better sequence. It moves the current candidate randomly in search for a better solution.

Step 4 [Update the best solution]:

Step 4.1: If the new solution is better than the current Black Hole (x_{BH}), then designate this new solution as new Black Hole (x_{BH}).

Step 4.2: Calculate the radius of the event of horizon (R) of the Black Hole clustering.

$$R = \frac{F_{BH}}{\sum_{i=1}^{Pop} F_i}$$

Where F_{BH} is the fitness for Black Hole clustering and F_i is the fitness of i^{th} clustering calculated using weighted fitness function calculated using Eq. (2).

Step 4.3: If a star enters this event horizon, it is absorbed by the Black Hole. It means, if $(F_{BH} - F_i < R)$, the clustering is discarded as it is regarded to be entered in event horizon of Black Hole and is thus vanished. Generate new clustering using Eq. (1) to balance the size of the population.

Step 5 [Genetic Algorithm]:

Step 5.1 [Input]: Take the population of candidate clustering from the Step 4 as input population of chromosomes and set parameters as shown in Table III. Calculate the fitness of each solution using function F described in Eq. (2).

Step 5.2 [Selection]: Select two parent chromosomes (tournament selection of size 2) from the population on the basis of their fitness.

Step 5.3 [Crossover]: Cross the parents selected in Step 5.2 to create new children and mutate new child at random positions in the chromosome.

Step 5.4 [Replace]: If this new offspring is better than the parents in terms of F calculated using Eq. (2), then accept it.

Step 6 [Output]: Output the candidate clustering having least value of combined objective function F .

Figure 2. Multi-Objective Weighted Black Hole Genetic Algorithm (MOWBHGA)

Step 1 [Initialize population]:

Step 1.1: Encode and initialize the population of possible clustering solutions. Set parameters as shown in Table III.

Step 1.2: Evaluate fitness of each candidate in the population using objective functions mentioned in Table II.

Step 1.3: Store the clustering that represent non-dominated vectors in the temporary repository (REP) and generate hyper-cubes to maintain best solutions.

Step 1.4: Select current best non-dominated clustering achieved so far and designates it as Black Hole (X_{BH}).

Repeat steps 2 to 6 until stopping criteria is met (as shown in Table III)

Step 2 [Identify new possible solutions]: For each iteration t , identify new location ($x_i(t+1)$) for each job sequence ($x_i(t)$) by using

$$x_i(t+1) = x_i(t) + rand * (x_{BH} - x_i(t))$$

Step 3 [Search for a better solution]: Evaluate fitness of each new clustering $x_i(t+1)$. If new candidate solution is better than the current candidate solution taking into consideration multiple objectives and their non-dominance, then replace the current solution with this new solution else ignore it. This step is required to locally search for a better sequence. It moves the current candidate randomly in search for a better solution.

Step 4 [Update the best solution]:

Step 4.1: If the new clustering $x_i(t+1)$ is better than the current Black Hole (x_{BH}), then designate this new clustering as new Black Hole (x_{BH}).

Step 4.2: Calculate the radius of event of horizon (R) of the Black hole clustering in non-dominated Pareto front by calculating components of radius on the basis of objectives mentioned in Table II. For each objective (h), the component of the radius is

$$R(h) = \frac{f_{BH}}{\sum_{i=1}^{Pop} f_i(h)}$$

Where f_{BH} is the fitness value of the Black Hole clustering and $f_i(h)$ is the fitness value of the h^{th} objective of i^{th} clustering. For the problem of software clustering under consideration, the number of objectives (h) is equal to 6, Pop is the size of the population under consideration (Table III).

Step 4.3: For each individual in the population and BH, if difference in fitness value of every corresponding objective function (h) dominates corresponding component of R i.e. $R(f_i(h) - f_{BH} \text{ dominates } R(h))$, the candidate clustering is discarded and a new star is generated randomly.

Step 5 [Apply Genetic algorithm]:

Step 5.1 [Input]: Take the population of candidate clustering from Step 4 as input population of chromosomes. Set parameters as shown in Table III. Calculate the fitness of each solution using the non-dominance approach on the basis of objectives mentioned in Table II.

Step 5.2 [Selection]: Select two parent chromosomes (tournament selection of size 2) from the population on the basis of their fitness.

Step 5.3 [Crossover]: Cross the parents selected in Step 5.2 to create new children and mutate new child at random positions in the chromosome.

Step 5.4 [Replace]: If this new offspring is better than the parents (non-dominating), then accept it.

Step 6 [Update best solutions]: Update hyper-cubes and REP to maintain current non-dominated clustering.

Step 7 [Output]: Return REP which includes resulting non-dominated clustering.

Figure 3. Multi-Objective Hybrid Black Hole Algorithm (MOBHGA)

TABLE IV. MOJOFM AND EDGESIM TO COMPARE MOWBHGA AND MOBHGA FOR SAMPLE ANDROID MOBILE APPLICATIONS

Mobile App	Approach	MoJoFM							EdgeSim						
		NP	NAE	NIE	NCP	NCD	NIP	Value	NP	NAE	NIE	NCP	NCD	NIP	Value
FourSquare	MOWBHGA	6	64	233	8	13	0	32.65306	6	57	240	10	13	0	61.27946
	MOBHGA	4	167	130	28	5	0	57.14286	4	167	130	28	5	0	65.29966
	Two-Archive	7	170	127	25	8	2	44.89796	7	262	35	23	0	1	64.53674
Sudoku	MOWBHGA	5	214	59	65	2	2	34.78261	5	214	59	65	2	2	59.34066
	MOBHGA	4	151	122	45	4	1	49.27536	2	220	53	56	1	0	67.39927
	Two-Archive	7	190	83	11	0	0	46.37681	7	165	108	12	0	0	62.27106
Apps Organizer	MOWBHGA	13	45	529	12	9	0	35.65892	13	45	529	12	9	0	57.49129
	MOBHGA	7	369	205	99	7	3	41.86047	13	100	474	16	0	1	63.67247
	Two-Archive	15	267	307	18	0	0	46.51163	15	400	174	17	0	0	62.71777
Punjab Kesari	MOWBHGA	45	160	5627	33	114	0	46.38429	45	138	5649	25	115	0	47.21347
	MOBHGA	32	1048	4739	400	102	6	49.09164	34	1513	4274	479	76	6	47.46846
	Two-Archive	15	400	174	17	0	0	48.72768	44	3233	2528	41	105	9	49.87098
Desi Coupon	MOWBHGA	4	845	32	238	0	2	59.3361	4	858	19	240	0	3	88.25542
	MOBHGA	3	406	471	137	12	0	60.16598	4	660	217	212	7	0	87.68529
	Two-Archive	5	380	497	67	0	1	59.30361	6	778	99	47	0	1	86.20297

Analyzing Table IV reveals that if MoJoFM is used as validation criteria, MOBHGA results in a highest value as compared to its counterparts in 4 out of 5 sample applications and if EdgeSim is used as validation criteria, MOBHGA results in a highest value in 3 out of 5 sample applications.

V. CONCLUSION

This work proposes the application of multi-objective BH algorithm and its hybrids with GA and auxiliary archive for clustering of mobile applications and the resulting modularizations are compared to Two-Archive algorithm. The results indicate that MOBHGA algorithm outperforms weighted objective based hybrid MOWBHGA and Two-Archive algorithm for clustering mobile applications. In future, PSO could be investigated for hybridizing Black Hole algorithm for obtaining even better clustering results.

REFERENCES

- [1] A. Hatamlou, "Black hole: A new heuristic optimization approach for data clustering," *Information Sciences*, vol. 222, pp. 175-184, 2013.
- [2] P. R. Srivastava, M. Chis, S. Deb, and X.-S. Yang, "An efficient optimization algorithm for structural software testing," *International Journal of Artificial Intelligence™*, vol. 8, pp. 68-77, 2012.
- [3] K. Srinivasan and D. Fisher, "Machine Learning Approaches to Estimating Software Development Effort," *Series on Software Engineering and Knowledge Engineering*, vol. 16, p. 52, 2005.
- [4] I. Hussain, A. Khanum, A. Q. Abbasi, and M. Y. Javed, "A Novel Approach for Software Architecture Recovery using Particle Swarm Optimization," *International Arab Journal of Information Technology (IAJIT)*, vol. 12, 2015.
- [5] A. S. Mamaghani and M. R. Meybodi, "Clustering of software systems using new hybrid algorithms," in *Computer and Information Technology, 2009. CIT'09. Ninth IEEE International Conference on*, 2009, pp. 20-25.
- [6] B. S. Mitchell and S. Mancoridis, "On the evaluation of the Bunch search-based software modularization algorithm," *Soft Computing*, vol. 12, pp. 77-93, 2008.
- [7] K. Praditwong, M. Harman, and X. Yao, "Software module clustering as a multi-objective search problem," *Software Engineering, IEEE Transactions on*, vol. 37, pp. 264-282, 2011.
- [8] K. J. Sullivan, W. G. Griswold, Y. Cai, and B. Hallen, "The structure and value of modularity in software design," in *ACM SIGSOFT Software Engineering Notes*, 2001, pp. 99-108.
- [9] K. Praditwong, "Solving software module clustering problem by evolutionary algorithms," in *Computer Science and Software Engineering (JCSSE), 2011 Eighth International Joint Conference on*, 2011, pp. 154-159.
- [10] C. Kishore and A. Srinivasulu, "Multi-objective approach for software module clustering," *International Journal of Advanced Research in Computer Engineering & Technology (IJARCET)*, vol. 2, 2012.
- [11] H. Abdeen, H. Sahraoui, O. Shata, N. Anquetil, and S. Ducasse, "Towards automatically improving package structure while respecting original design decisions," in *Reverse Engineering (WCRE), 2013 20th Working Conference on*, 2013, pp. 212-221.
- [12] A. Ibrahim, D. Rayside, and R. Kashef, "Cooperative based software clustering on dependency graphs," in *Electrical and Computer Engineering (CCECE), 2014 IEEE 27th Canadian Conference on*, 2014, pp. 1-6.
- [13] K. Jeet and R. Dhir, "Software Architecture Recovery using Genetic Black Hole Algorithm," *ACM SIGSOFT Software Engineering Notes*, vol. 40, pp. 1-5, 2015.
- [14] W. Mkaouer, et al., "Many-Objective Software Remodularization Using NSGA-III," *ACM Transactions on Software Engineering and Methodology (TOSEM)*, vol. 24, p. 17, 2015.
- [15] R. T. Marler and J. S. Arora, "The weighted sum method for multi-objective optimization: new insights," *Structural and multidisciplinary optimization*, vol. 41, pp. 853-862, 2010.
- [16] C. A. C. Coello, G. T. Pulido, and M. S. Lechuga, "Handling multiple objectives with particle swarm optimization," *Evolutionary Computation, IEEE Transactions on*, vol. 8, pp. 256-279, 2004.
- [17] F. Beck and S. Diehl, "On the impact of software evolution on software clustering," *Empirical Software Engineering*, vol. 18, pp. 970-1004, 2013.

MABT - a multiagent-based toolkit for transforming existing systems into self-adaptive systems

Lu Wang, Qingshan Li*, Yishuai Lin, Hua Chu
 Software Engineering Institute
 Xidian University
 Xi'an, P. R. China
 qshli@mail.xidian.edu.cn

Abstract—Some methods and auxiliary platforms/tools have been proposed to design and develop Self-Adaptive Systems (SASs). However, these methods and tools ignore how to turn existing systems to SASs. Therefore, the paper proposes a multiagent-based toolkit for adding self-adaptive abilities to the existing systems. With the Agent Packager and Rule Designer in it, existing systems are transformed into Multi-Agent Systems (MAS), without modifying codes. Based on the Supporting Platform in MABT, MAS run normally and achieves self-adaption with both Global and Local approach.

Keywords—self-adaptive system; software transformation; multi-agent system; software reuse; software development kit

Since the cost of developing new software systems is prohibitive, there are many researches on the existing software systems. The complexity of adapting the existing softwares to software changes (such as the abnormal software units, the changes of requirements, and the modifications of software architecture) is beyond the capabilities of manual control, so it is badly needed to make the existing software become the SASs which are able to measure and spontaneously adjust the behaviors based on the analysis of software changes.

Therefore, this paper proposes a MultiAgent-Based Toolkit, called MABT, for transforming existing software into SASs. In this toolkit, the agent technology (i.e. the agent based framework, the design patterns of SASs based on agents, agents' role modeling of SASs, etc.) is used to establish the SASs. The proposed MABT is divided into three phases, as shown in figure 1. The Toolkit offers Agent Packager and Rule Designer for software developers in the design phase. The original software units are packaged as agents by Agent Packager with the Agent Model and the Structure of Packaging. And the organizations of and collaborations among units are transferred into collaborations among agents and defined by Rule Designer with the Agent-based Logic Description Language [2] and the Comprehensive Checking Mechanism. With these tools, developers can quickly transform the existing software into MAS. And, new software units which can enhance the adaptive ability of the software can be added to MAS by these toolkit in the design phase. In the operational phase, toolkit can support the normal running and self-adaption of MAS by Supporting Platform. Basic Services supports the normal running of MAS mainly by providing the services of communications and managements. Self-adaptive Services are responsible for

providing the supports for self-adaptation in both Global and Local self-adaptive approaches. For Global Events which influence many agents (such as the changes of the context or requirements), the Global approach can adjust the collaborations among agents to form a new behavior of software. For Local Events which influence only an agent, Local approach adjusts the external services provided by an agent to the adjustment of the Business Logics.

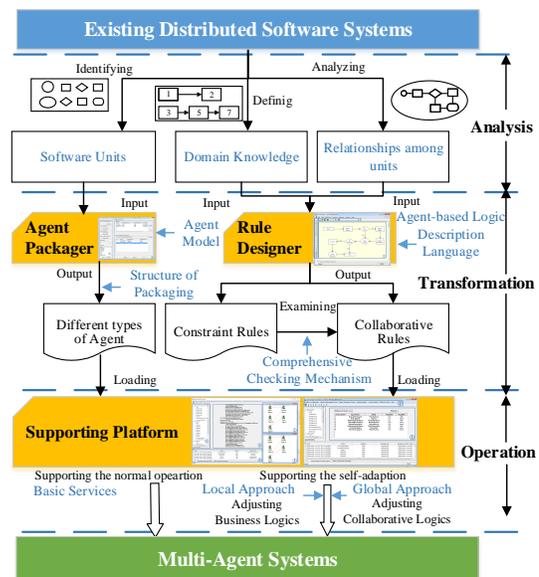


Figure 1. The structure of the MABT

ACKNOWLEDGMENT

This work is supported by the Projects (61373045, 61173026, 61303031) supported by the National Natural Science Foundation of China; Projects (BDY221411, K5051223008, JB151003) supported by the Fundamental Research Funds for the Central Universities of China.

REFERENCES

- [1] L. Qingshan, C. Hua, and L. Cong, "An Agent-based Logic Description Language for Dynamic Evolution," Third World Congress on Software Engineering, Xinhua Huang, Li Da Xu, Eds. IEEE, 2012, pp. 15-18, November 2012.

An Efficient Algorithm to Identify Minimal Failure-Causing Schemas from Exhaustive Test Suite

Yuanchao Qi¹, Qi Wang¹, Chiya Xu¹, Tieke He², and Ziyuan Wang^{1*}

¹School of Computer Science and Technology, Nanjing University of Posts and Telecommunications, Nanjing, China

²State Key Lab for Novel Software Technology, Nanjing University, Nanjing, China

*Corresponding: wangziyuan@njupt.edu.cn

Abstract—Combinatorial testing is widely used to detect failures caused by interactions among parameters for its efficiency and effectiveness. Fault localization plays an important role in this testing technique. And minimal failure-causing schema is the root cause of failure. In this paper, an efficient algorithm, which identifies minimal failure-causing schemas from existing failed test cases and passed test cases, is proposed to replace the basic algorithm with worse time performance. Time complexity of basic and improved algorithms is calculated and compared. The result shows that the method that utilizes the differences between failed test cases and passed test cases is better than the method that only uses the sub-schemas of those test cases.

Keywords—Combinatorial testing, fault localization, minimal failure-causing schema, algorithm.

I. INTRODUCTION

Softwares may be affected by the interactions among its parameters. These interactions need to be tested to guarantee the quality of software. But for the software with k parameters, it is unacceptable to cover all the possible k -tuple combinations of parametric values. Combinatorial testing provides a trade-off between the testing cost and the degree of combinatorial coverage. It has been widely used for its efficiency and effectiveness, especially, in highly-configurable systems [1]. Test case generation and fault localization based on failure-causing schemas are hot areas of research.

Many researches focus primarily on generating test cases to filter suspicious failure-causing schemas. But there are few materials to discuss how to filter them. The default option is to construct all possible suspicious failure-causing schemas from failed test cases and filter them by passed test cases. In this paper, we propose an improved algorithm with better time performance. The analysis of time complexity shows the advantage of proposed algorithm. And the efficient algorithm has been applied in our practice to compute minimal failure-causing schemas for boolean-specification testing and Siemens program suite.

II. BACKGROUND

The model of minimal failure-causing schema was proposed by Nie et. al [2].

For a software system with k parameters, we suppose each parameter f_i has a_i ($1 \leq i \leq k$) discrete valid values. Let $F = \{f_1, f_2, \dots, f_k\}$ denote the set of parameter, and $V_i =$

$\{0, 1, \dots, a_{i-1}\}$ ($i = 1, 2, \dots, k$) the value set for f_i without loss of generality.

Definition 1. (Schema). A k -tuple $s = (-, \dots, -, v_{i_1}, -, \dots, -, v_{i_2}, -, \dots, -, v_{i_\tau}, -, \dots, -)$ is a schema with strength τ , or a τ -way schema (or τ -schema for short) ($1 \leq \tau \leq k$). Where τ values are fixed as $v_{i_1} \in V_{i_1}, v_{i_2} \in V_{i_2}, \dots, v_{i_\tau} \in V_{i_\tau}$, and other $k - \tau$ values are not fixed and represented as “-”.

Definition 2. (Sub-schema and parent-schema). Schemas $s_1 = (v_1, v_2, \dots, v_k)$ and $s_2 = (v'_1, v'_2, \dots, v'_k)$ are τ_1 -schema and τ_2 -schema respectively ($\tau_1 \leq \tau_2$). If $\forall 1 \leq i \leq k, (v_i = -) \vee (v_i = v'_i)$ is true, then s_1 is a sub-schema of s_2 , and s_2 is a parent-schema of s_1 . It is denoted as $s_1 \prec s_2$. Especially, if $s_1 \neq s_2$, then s_1 is a real sub-schema of s_2 , and s_2 is a real parent-schema of s_1 .

Definition 3. (Failure-causing schema). A schema s is a failure-causing schema (or *FS* for short), if $\forall t \in T_{all} = V_1 \times V_2 \times \dots \times V_k, s \prec t \Rightarrow t$ is failed test case.

Definition 4. (Minimal failure-causing schema). A failure-causing schema s is a minimal failure-causing schema (or *MFS* for short), if any real sub-schema of s is not a failure-causing schema.

III. ALGORITHMS

People pay more attention to the problem that how to identify minimal failure-causing schemas accurately. However, efficiency is also a fundamental issue. In this section, we will introduce the most used basic algorithm and then propose an improved one with better time complexity.

A. Basic algorithm

The basic algorithm, which identifies minimal failure-causing schemas from failed test cases and passed test cases, was mentioned in many materials. But its detailed process was often omitted. Here we describe the basic algorithm and analyze its time performance.

For each failed test case t , there are C_k^1 1-way sub-schemas, C_k^2 2-way sub-schemas, ..., and C_k^k k -way schemas.

When filtering suspicious schemas, from $nf \times C_k^1$ 1-way sub-schemas of failed test cases will be filtered by $np \times C_k^1$ 1-way sub-schemas of passed test cases, to $nf \times C_k^k$ k -way sub-schemas of failed test cases be filtered by $np \times C_k^k$ k -way sub-schemas.

When comparing two i -way schemas ($i = 1, 2, \dots, k$), the values of i parameters should be compared. Therefore, the total

Algorithm 1: Identify MFS using failed and passed test cases

Input: $FTCS$: set of failed test cases

$PTCS$: set of passed test cases

Output: $MFSs$: set of minimal failure-causing schemas

1. $FSS = \emptyset$;
 2. **For Each** failed test case $t \in FTCS$
 3. $FSS = FSS + SubScheSet(t)$;
 4. **End For**
 5. **For Each** passed test case $t \in PTCS$
 6. $FSS = FSS - SubScheSet(t)$;
 7. **End For**
 8. $MFSs = \{s | s \text{ is minimal schemas in } FSS\}$;
-

time complexity of filtering suspicious schemas in Algorithm 1 should be:

$$O(np \times C_k^1 \times nf \times C_k^1 \times 1 + np \times C_k^2 \times nf \times C_k^2 \times 2 + \dots + np \times C_k^k \times nf \times C_k^k \times k) \sim O(np \times nf \times \sum_{i=1}^k (i \times (C_k^i)^2)).$$

Additionally, in the process of selecting minimal ones from the set of failure-causing schemas, we can filter τ -way failure-causing schemas by $(\tau - 1)$ -way's ($\tau = 2, 3, \dots, k$), for each failed test case. So there are totally $C_k^k \times C_k^{k-1} + C_k^{k-1} \times C_k^{k-2} + \dots + C_k^{k-2} \times C_k^1$ parametric values should be checked for each failed test case. Here note that $O(\sum_{i=2}^k (C_k^i \times C_k^{i-1})) \sim O(\sum_{i=2}^k (C_k^i)^2) \sim O(\sum_{i=1}^k (C_k^i)^2) \sim O(C_{2k}^k)$.

Therefore, the time complexity of the whole Algorithm 1 should be: $O(np \times nf \times \sum_{i=1}^k (i \times (C_k^i)^2) + nf \times C_{2k}^k)$.

B. Improved algorithm

Factually, the process of extracting and filtering suspicious schemas in the basic algorithm could be optimized to enhance its time performance. We will propose an improved algorithm by utilizing the differences between failed test cases and passed test cases.

Considering a failed test case t and a passed test case t' , we could construct a set of parameters $Diff_Param(t, t')$ that contains all parameters whose parametric values in t and t' are different. So, the process to identify MFS is described in Algorithm 2.

Algorithm 2: Identify MFS using failed and passed test cases

Input: $FTCS$: set of failed test cases

$PTCS$: set of passed test cases

Output: $MFSs$: set of minimal failure-causing schemas

1. $FSS = \emptyset$;
 2. **For Each** failed test case $t \in FTCS$
 3. $Diff(t) = \emptyset$;
 4. **For Each** passed test case $t' \in PTCS$
 5. $Diff(t, t') = \{f_i \in F | t[i] \neq t'[i]\}$;
 6. $Diff(t) = Diff(t) + \{Diff(t, t')\}$;
 7. **End For**
 8. $FSS(t) = \{s \in SubScheSet(t) |$
 for each $Diff(t, t') \in Diff(t), \exists f_i \in$
 $Diff(t, t') \text{ that } s[i] \neq -\}$;
 9. $FSS = FSS + FSS(t)$;
 10. **End For**
 11. $MFSs = \{s | s \text{ is minimal schemas in } FSS\}$;
-

For a failed test case t , there are k parametric values which should be checked when constructing a $Diff_Param(t, t')$ with the passed test case t' . If there are np passed test cases, it is $np \times k$. So there are totally $np \times nf \times k$ parametric values which should be checked when constructing these sets for all failed test cases.

For a failed test case t , when selecting its failure-causing sub-schemas, there are np different $Diff_Param(t, t')$ to be checked. So there are totally $np \times \sum_{i=1}^k (i \times C_k^i)$ parametric values which should be checked for one failed test case, and totally $np \times nf \times \sum_{i=1}^k (i \times C_k^i)$ parametric values should be checked for all failed test cases.

Therefore, the total time complexity of selecting all failure-causing schemas in Algorithm 2 should be:

$$O(np \times nf \times k + np \times nf \times \sum_{i=1}^k (i \times C_k^i)) \\ \sim O(np \times nf \times \sum_{i=1}^k (i \times C_k^i))$$

Since the time complexity of selecting minimal failure-causing schemas is $O(C_{2k}^k)$ for each failed test case, the time complexity of the whole Algorithm 2 should be: $O(np \times nf \times \sum_{i=1}^k (i \times C_k^i) + nf \times C_{2k}^k)$.

IV. DISCUSSION

A. Outputs of Two Algorithms

Algorithm 1 and Algorithm 2 obtain the same outputs for the same inputs. It is clear in the description of two algorithms, especially in the description of improved one.

B. Comparing Time Performance

According to the binomial theorem, $C_{2k}^k = \sum_{i=1}^k (C_k^i)^2$. Since there are:

$$\sum_{i=1}^k (C_k^i)^2 < \sum_{i=1}^k (i \times (C_k^i)^2) \\ \sum_{i=1}^k (i \times C_k^i) < \sum_{i=1}^k (i \times (C_k^i)^2)$$

So it is obvious that $O(np \times nf \times \sum_{i=1}^k (i \times C_k^i) + nf \times C_{2k}^k) < O(np \times nf \times \sum_{i=1}^k (i \times (C_k^i)^2) + nf \times C_{2k}^k)$. It means that the time complexity of Algorithm 2 is less than that of Algorithm 1. Then we can make a conclusion that Algorithm 2 is better than Algorithm 1.

V. CONCLUSION

In this paper, we carefully study two algorithms that could identify MFSs by utilizing failed test cases and passed test cases. The time complexity of two algorithms shows that the approach which utilizes the difference between every failed test case and passed test case is clearly better than the other one. We believe that the research will improve the effectiveness and efficiency of practical testing.

REFERENCES

- [1] C. Nie, H. Leung. A survey of combinatorial testing. ACM Computing Surveys (CSUR), 2011, 43(2): 11.
- [2] C. Nie, H. Leung. The Minimal Failure-causing Schema of Combinatorial Testing. ACM Transactions on Software Engineering and Methodology (TOSEM), 2011, 20(4): 15.

A qualitative analysis of the adherence between the Information Technology Solution Acquisition Guide, for Brazilian Federal Public Administration and, the CMMI models

Luiz S.P. Silva, Alexandre
M.L. Vasconcelos Federal
University of Pernambuco
(UFPE)
Recife - PE - Brazil
lsps@cin.ufpe.br,
amlv@cin.ufpe.br

Renata T. Moreira
Federal University of Lavras
(UFLA)
Lavras – MG - Brazil
renata@dcc.ufla.br

Maurício R. A.
Souza
Federal University
of Minas Gerais
Belo Horizonte –
MG – Brazil
mrasouza@dcc.ufmg.br

Suzana C. B. Sampaio
University Federal Rural of
Pernambuco (UFRPE)
Recife - PE - Brazil
suzana@deinfo.ufrpe.br

Abstract — quality models and standards guide Software Process Improvement Initiatives. Usually aiming to establish the best practices to instruct the definition of processes and support organizational assessment of the maturity and capability level. However, despite these initiatives, the best practices application in Brazilian public organizations is impaired by various obstacles regarding the process of Contracting IT solutions by the Federal Public Administration (APF), the main contractor of software and services in Brazil. Among these obstacles the ones that stand out are process complexity and the continuous supervision of control bodies. In order to minimize these obstacles, the court Union Accounts (TCU), recommended the establishment of the Normative Instruction SLTI/MPOG 04/2014, containing guidelines for the acquirement of IT Solutions and a Guide for Contracting IT Solutions (GCSTI). This work aims at identifying the maturity and adherence of GCSTI relative to CMMI models. To achieve this goal, defined and carried out a method of mapping between GCSTI and CMMI models. The survey results show the maturity and adherence to CMMI models based on procedures defined in GCSTI.

Keywords: Software and Services Process Improvement, Process IT Solutions Contracting, CMMI models.

INTRODUCTION

The advances and increasing dependence of Information Technology Services (IT) (which include care services, information storage, data, customer support and the various resources and technological means available to society) caused service providers to struggle at maintaining the expected service quality required by clients (LIRA et al, 2008; BRICKLEY, 2001; BERGAMASCHI, 2004). Researchers and practitioners investigate the factors that affect the provision of IT services to meet the needs and expectations of customers (CRUZ et al., 2011, SILVA, 2013).

The Brazilian Federal Public Administration (APF) is the largest consumer and buyer of IT products and services in Brazil (SILVA, 2013). Its procurement process for contracting IT services and solutions is guided by the Normative instructions IN / SLTI / MPOG 04/2014 and the Guide on Contracting IT Solutions – the GCSTI. However, from the providers perspective, meeting these recommendations is complex, and may incur in additional expenditure for achieving compliance.

Therefore, this study aims to evaluate the process of Contracting IT solutions based on CMMI models, in order to analyze what is the maturity required for execution of the APF procurement process. We mapped the GCSTI recommendations to the good practices provided by the CMMI models “CMMI for development” (CMMI-DEV), CMMI for Service (CMMI-SVC), and CMMI for Acquisition (CMMI-ACQ).

This paper is organized as follows: Section I presents the theoretical framework; Section II presents the mapping between the GCSTI and CMMI models, Section III the challenges founds, and Section VI, presents conclusions the Work Completion.

I. THEORETICAL FRAMEWORK

The normative instruction IN/SLTI/MPOG 04/2014 (SLTI, 2015) provides governance mechanisms for contracting IT services and solutions. To support the compliance to the IN/SLTI/MPOG 04/2014, the Good Practices Guide on Contracting IT Solutions (GCSTI) was created as a compilation of process, activities and tasks executed during the procurement process (CAVALCANTI, 2015). It is composed of three phases: (i) IT Solution Acquisition Plan; (ii) Selection of IT Solution Provider; and (iii) IT Solution Contract Management.

The CMMI Model (Capability Maturity Model Integration) is a collection of components from several maturity models and an assessment process maintained by the CMMI Institute (GALLAGHER et al., 2010; CHRISSIS et al., 2010; FORRESTER et al., 2010). These components are grouped in constellations, each one providing guidelines for process improvement in a given area of interest, such as acquisition (ACQ), development (DEV) and services (SVC).

II. MAPPING BETWEEN THE GUIDE OF IT SOLUTIONS CONTRACTING AND MODELS CMMI

In order to provide an assessment of the required organization maturity for providing IT services and solutions in compliance with the GCSTI, we mapped the GCSTI recommendations to the CMMI practices. The mapping was executed in nine steps: (i) Study the models; (ii) Delimitation of the work scope; (iii) Definition of the classification criteria; (iv) Establishment of initial Mapping Form; (v) Establishment of the Standard Form for Mapping; (vi) Establishment of the Analysis Standard Form; (vii) Comparing Models; (viii) Consolidation of Results; and (ix) Validation with the Peer Review Technical.

We evaluated the compliance of the GCSTI processes to Specific Practices from the CMMI models. Due to space restriction,

Table 1 provides only a summary of the mapping. For each phase of the GCSTI process (IT Solution Acquisition Plan, Selection of IT Solution Provider, and IT Solution Contract Management) we evaluated the compliance with the Specific Practices from each of the CMMI Models (DEV, SVC and ACQ) Practice Areas.

Some important aspects were identified in the final consolidation of the results of the mapping between MCTI model and CMMI models. The core Process Areas (common to the three models) had the same percentage of attendance by GCSTI model processes

Table 1. Mapping Consolidation

Process Areas	CMMI ACQ	CMMI DEV	CMMI SVC
PP	100%	100%	
WP			100%
PMC	100%	100%	
WMC			100%
CM	85,71%	85,71%	85,71%
PPQA	100%	100%	100%
MA	68,75%	68,75/	68,75/
SAM		100%	100%
REQM	80%	80%	80%
SD			100%
OPF	11,11%	11,11%	11,11%
OPD	50%	50%	50%
DAR	100%	100%	100%
OT	28,57%	28,57%	28,57
IPM	70%	70%	
IWM			70%
RSKM	85,71%	85,71%	85,71%
ARD	100%		
AM	100%		
AVER	87,50%		
AVAL	100%		
SSAD	100%		
ATM	100%		
RD		100%	
TS		100%	
PI		100%	
VER		87,50	
VAL		100%	
CAM			16,67%
IRP			50%
SSD			91,67%
SCON			75%
SST			60%
STSM			50%

Based on these results, it is noted that there is a deficiency in the sequence of execution of GCSTI model processes with respect to CMMI models. Whereas the GCSTI model serves various process areas of CMMI models in different maturity levels. While that process areas related to engineering Acquisition, Development and Services are attended by almost 100% (Maturity Level 3), the process areas from Maturity Level 2, which defines the “managed level”, is not attended completely. Therefore, the areas related to level 3 maturity, are not being performed after the meeting the level 2 process areas maturity, which can result in problems and difficulties in implementing the processes.

The mapping between CMMI Models and GCSTI process showed that, although it has a high adherence to CMMI Process Areas, the GCSTI process does not provide a sequential or staged implementation suggestion. Thus some practices from CMMI

Maturity Level 2 are not completed attended, and it may lead to difficulty in introducing practices from higher maturity levels.

III. CONCLUSIONS

The results shows that there is a lack of alignment between the GCSTI process and the methodologies, norms and models commonly applied as quality standards in the software and service industry. The CSGTI defines process and activities equivalent to CMMI Maturity Level 3. In this Maturity Level, organizations already have processes defined for managing projects, services and acquisition. Therefore, there is a challenge in elevating the maturity of IT organizations in Brazil, promoting their capacity in order to become ready for providing solutions for the government. By achieving this, the brazilian government will promote small companies, empowering the software industry and ensuring constitutional principles on the IT contracts.

IV. REFERENCES

ABES - Brazilian Association of Software Companies. 2015. Available at: <<http://www.abes.org.br/>>. Accessed: 08/02/2016.

ABREU, M. F. The risks of IT outsourcing and the adoption of new IT and its relations with the risks to the competitive strategies of organizations. 2009.

BERGAMASCHI, Sydney. Models for the Management of Outsourcing Information Technology: An Exploratory Study. Thesis (doctoral) - University of São Paulo, 2004.

Bernstorff, V. H; CUNHA, J. C. O. organizations to seek and achieve with the outsourcing of information technology In: XXIII Annual Meeting of ANPAD 1999, Foz do Iguauçu / PR ANAL. ANPAD 1999.

BRAGA, R. Audit of IT Governance. Brasília: TCU / ISC 2009.

CAVALCANTI, S.C. The New Model IT Solutions Contracting for the Federal Public Administration. 2a. Ed. Belo Horizonte. Publishing Forum 2015.

CHRISSIS, M.D., KONRAD, M. AND S. SHRUM "CMMI: guidelines for process integration and product improvement. " Addison-Wesley. 2010.

CRUZ, C. S. of. IT Governance and Legal Compliance in the Public Sector: A Framework Reference Normative for IT Service Contracts. 2008. Master's Thesis. Catholic University of Brasilia, Brasilia, 2008. Available at: <[Http://www.btdt.ucb.br/tede/tde_arquivos/3/TDE-2008-11-25T123713Z-687 / Public / TextoCompleto Cruz - 2008.pdf](http://www.btdt.ucb.br/tede/tde_arquivos/3/TDE-2008-11-25T123713Z-687 / Public / TextoCompleto Cruz - 2008.pdf)>. Accessed: 15/12/2016.

FORRESTER, E., Buteau, B., Shrum, S. CMMI: Guidelines exceeds Service. Addison-Wesley. 2010.

GALLAGHER, B., PHILLIPS, M., RICHTER, K., Shrum, S. CMMI: Guidelines for Improving the Acquisition of Products and Services. Addison-Wesley. 2010.

ISO / IEC, 2011. International Organization For Standardization / International Electrotechnical Commission. ISO / IEC 20000 Information technology- Management Service, Geneve: ISO 2011.

LIRA, W. S. ; CANDID, G. A. ; ARAUJO, G. & M. BARROS, M. A. The search and the use of information in organizations. Perspectives on Information Science. Vol. 13, no. 1, Belo Horizonte, 2008.

SEI, 2010. Software Engineering Institute. CMMI for Development. 2010.

SILVA, L. S. P. Model of IT Solutions Contracting: A Comparative Analysis to Identify the maturity and adherence to CMMI-ACQ models, CMMI-DEV and CMMI-SVC. 2013. Dissertation (Master in Computer Science) - Computer Center - Federal University of Pernambuco, Recife, Pernambuco, 2013.

Early Detection of Suicide Using Big-Data Analytics in Real Time

Hardik A. Patel, Cheng-Yuan Hsieh
Knowledge Systems Institute

Abstract—This paper presents an application, wherein predictive analysis is used to churn social media comments and posts for identifying individuals on these sites susceptible to suicidal thoughts and tendencies. So an early intervention could be made saving lives of such individuals. Also presented is a model wherein a branch of big data analytics and sentimental analytics has been used to identify profiles of users with suicidal posts.

Index Terms—Big data, Real-time, Predictive analysis .

I. INTRODUCTION

This application is meant to be used by suicide prevention agencies. The growth of social media has presented a new set of challenges and opportunities in the field of suicide prevention. Studies have shown that many people have presented their feeling related to depression on social media [1]. A timely intervention could save lives of people who might have not given a thought of getting help themselves by going to counseling services offered by these agencies. American Foundation for Suicide Prevention (AFSP) has a webpage dedicated to warnings of suicide [2], or indications where a person might be showing suicidal tendencies. Research has shown that suicidal ideation or self-injurious behavior is an early warning of suicide [3]. American Association of Suicidology (Suicidology.org), lists vocalizing suicidal thoughts as one of the primary warning indicators for suicide. This application will provide assistance to organizations by fetching information from social media websites like Facebook, Twitter etc., in real time when posts and/ or comments exhibiting such tendencies are made.

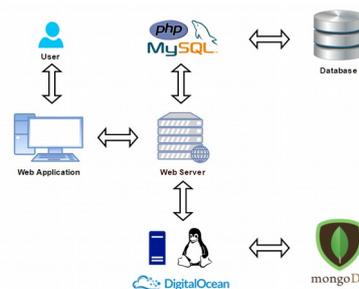
II. RESEARCH METHODOLOGY

The impact of media on suicidal behavior seems to be most likely when a method of suicide is specified in detail. Sometimes newspapers or other media outlets like radio, television or the internet publish suicide stories or reports in details as to how it happened and what methods were used in the suicide. Suicide prevention agencies, both federal state and private charities, have been very active in helping individuals with depression and suicidal behavior. The main drawback of conventional systems is timely intervention. All of the conventional systems relied primarily to individuals seeking out help for themselves or someone they know. All other cases where the primary warning signs are not detected or the individual is hesitant to seek out help, these agencies with all their excellent resources would not be able to provide any help. The application will identify such posts based upon certain data sets. These data sets comprise of certain words/phrases which help the application identify such profiles by pointing to the associated analysis would provide suicide prevention agencies

a list of such subjects who might need help/intervention to combat through their depression and anxieties associated with suicide victims.

III. SYSTEM OVERVIEW

The system has a multi-tier Architecture and includes web server, a PHP MySQL server, a Linux server hosted on Digital Ocean cloud ecosystem and a MongoDB [4] database for storing and retrieving the tweeter feed in JSON format. The live twitter feeds are fed from the MongoDB database on the Linux server provided by Digital Ocean Cloud infrastructure provider[5]. Sentiment analysis on the twitter feeds takes place in the Linux server and the resulting output is stored in the MongoDB database in JSON format. The web application accesses this information and populates it on the Home Screen. An agent can see twitter feeds which are captured in real-time, where as a supervisor can see the tweets flagged by the Agents. Then a timely intervention could be issued.



IV. CONCLUSION

This is a tool for the suicide prevention agencies who can use it to do predictive analysis using big data, in identifying individuals suffering from suicidal tendencies, thereby helping them to make a timely intervention and saving lives.

V. References

- [1] K. M. M. N. M. E.-F. M. & S. M. Becker, "Parasuicide online: Can suicide websites trigger suicidal behaviour in predisposed adolescents?," *Nordic Journal of Psychiatry*, vol. 58, pp. 111-4, 2004.
- [2] "American Foundation for Suicide Prevention," [Accessed 11 01 2016] <https://www.afsp.org/understanding-suicide/suicide-warning-signs>.
- [3] R. D. Goldney, "Suicide Prevention," Oxford University Press, 2008, p. 11.
- [4] Tutorials Point, "Mongo DB," [Accessed 09 March 2016] http://www.tutorialspoint.com/mongodb/mongodb_overview.htm.
- [5] Databricks, "Apache Spark," Databricks, [Accessed 31 January 2016] <https://databricks.com/spark/about>.

DOI reference number: 10.18293/SEKE2016-251

Authors' Index

Abran, Alain	357
Acuña, Silvia T.	537
Adam, Sebastien	357
Affonso, Frank	617
Agilar, Everton	345
Al Balushi, Tasira	341
Al-Najdi, Atheer	454
Albuquerque, Adriano	402, 575
Ali, Saqib	341
Almeida, Hyggo	8, 81, 267
Almeida, Rodrigo	345
Amorim, Anderson R.	555
An, Ran	219
Anu, Vaibhav	531
Aparecida Martinez Zaina, Luciana	231
Aquino, Gibeon	551
Arruda, Filipe	420
Arumugam, Chamundeswari	565
B. P., Gautham	197
Bacelo, Ana Paula	628
Balby Marinho, Leandro	644
Barbosa Araújo Ramos, Felipe	267
Barros Sampaio, Suzana Cândido de	657
Barros, Flavia	420
Barroso, Jonatas	575
Bennin, Kwabena Ebo	527
Bhattasali, Tapalina	93
Bissyandé, Tegawendé F.	273
Bosu, Michael Franklin	527
Bourreau, Eric	105
Bradshaw, Gary	531
Briot, Jean-Pierre	369
Bublitz, Frederico Moreira	434
Bueno Ruas Oliveira, Lucas	363
Campos Pereira, Wenderson	466
Canedo, Edna	345
Carver, Jeffrey	531
Castro, John W.	537
Cavalli, Ana	408
Cerqueira, Thaciana G. O.	644
Chaki, Nabendu	93
Chaki, Rituparna	93

Cham, Brian	443
Chang, Shikuo	183
Chateau, Annie	105
Chaves E Silva, Lenardo	414
Chawla, Shaurya	386
Che, Meiru	486
Chen, Daniel	492
Chen, Guangqun	120
Chen, Kai	147, 191
Chen, Liqiong	439
Chen, Mingsong	547
Chen, Xiangping	141, 589
Chen, Xiaohong	547
Chen, Xu	213
Chen, Zhenyu	243
Cheng, Ming	171
Chu, Hua	654
Chung, Lawrence	569
Ciancarini, Paolo	249
Colaço Júnior, Methanias	466
Conte, Tayana	595, 600
Contessoto, Allan G.	555
Corral, Luis	543
Costa, Ingrid	595
Costa, Túlio	434
Da Costa Santos, Leonardo	414
Da Silva, Raissa	8
Da Silveira, Marcos	329
Dai, Ruxin	219
Dash, Debasis	99
Dash, Juthika	99
de Borba Campos, Marcia	628
de Macedo Rodrigues, Elder	628
Dharia, Saumil	386
Dhir, Renu	650
Diallo, Nafi	605
Dias-Neto, Arilo Claudio	317
Ding, Junhua	390
Ding, Yong	46
Diniz de Souza, Adler	611
Do, Quan	486
Dobbie, Gillian	443
Dong, Xiang	147, 191
Dongdong, Cheng	498
Drira, Khalil	329
Du, Huiying	110
Du, Yiman	279

Eckert, Candice	443
Eirinaki, Magdalini	386
El Ioini, Nabil	543
Esteca, Antonio M. N.	555
Exposito, Ernesto	329
Eyal-Salman, Hamzeh	1
Fan Chiang, Sung-Ju	492
Fan, Guisheng	225, 439
Far, Behrouz	301
Faria, João	515
Feng, Fuli	177
Ferdjouxh, Adel	105
Filho, Jackson Feijó	317
Fontão, Awdren	317
França, Joyce	323
Freire, Arthur	8, 81
Fronza, Ilenia	543
Fu, Mandi	237
Fu, Yujian	623
Gadelha, Bruno	595
Gao, Jerry	99, 375, 480
Gao, Kun	52, 120
Gao, Xin	110
Garcia-Martinez, Ramon	504
Garrido, José Luis	569
Ghardallou, Wided	605
Gois Costa, Juli Kelle	466
Gontijo Tavares, Breno	611
Gonçalves, Rafael	153
Gresse von Wangenheim, Christiane	153
Guessi, Milena	363
Gupta, Sanika	638
Hao, Rui	462
He, Tieke	243, 462, 655
He, Wenjing	75
He, Xudong	623
Heim, Robert	351
Hong, Jang-Eui	569
Hong, Liang	213
Hsieh, Cheng-Yuan	659
Hu, Wenhua	531
Hu, Ye	52, 120
Huang, Ailing	297
Huang, Ruozi	589

Huang, Weizhi	87
Hui, Darren	486
Höst, Martin	64
Ibershimi, Aristeia	543
Ibrahim, Hamdy	301
Jain, Vijesh	386
Jeet, Kawal	650
Jia, Xuesong	124
Jia, Yuhan	279
Jiang, Jing	177
Jiang, Min	171
Jinlong, Huang	498
José Pereira de Lucena, Carlos	369
Kaliamourthy, Baskaran	565
Kamthan, Pankaj	579
Keung, Jacky	527
Khayati, Oualid	14
Klein, Jacques	273
Kraeim, Naoufel	14
Kulesza, Uirá	583
Kung, David	638
Le Traon, Yves	273
Leite, Gustavo	617
Li, Daoyuan	273
Li, Huakang	124
Li, Li	128
Li, Ning	87
Li, Qingshan	654
Li, Xiumin	46
Lian, Xiaoli	177
Libo, Zhang	31
Lijun, Yang	498
Lima, Bruno Rafael B.	510
Lima, Josimar	323
Lin, Jin	255
Lin, Jun	75
Lin, Lan	309
Lin, Mouguang	141
Lin, Pengpeng	219
Lin, Yang	31
Lin, Yishuai	654
Lins de Vasconcelos, Alexandre Marcos	657
Lins Rodrigues, Rodrigo	510
Liu, Chao	134

Liu, Haijun	297
Liu, Jia	243, 462
Liu, Jin	213, 237
Liu, Tao	203
Liu, Yan	255, 261, 634
Liu, Yunxiang	439
Long, Yonghao	589
Lopes, Adriana	595
Lopes, Andrei G.	434
Lorayne, Anne	267
Lou, Xin	261
Luisa Medeiros, Ana	414
Luo, Fei	225
Luz, Helder J. F.	396
Ma, Chao	141
Ma, Chuanxiang	237
Ma, Yutao	159
Machado, Crystiano José Richard	510
Maciel, Alexandre	510
Magalhães, Cláudio	470
Magües, Daniel A.	537
Maia, Eliot	470
Malhotra, Chetan P	197
Mani, Nariman	333
Martins, Sebastian	504
Melo, Mário	551
Menezes, Jislane	323
Menezes, Rinaldo V.	434
Mensah, Solomon	527
Mezghani, Emna	329
Miao, Chunyan	75
Mikkonen, Tommi	114
Mili, Ali	605
Mo, Wenkai	87
Mondal, Amit Kumar	521
Moore, Mary Frances	474
Moraes Do Nascimento, Nathalia	369
Moreira, Renata	657
Mota, Alexandre	470
Moura Costa, Antonio Alexandre	267
Nakagawa, Elisa	617
Nascimento, Andre	466
Nebut, Clémentine	105
Nguyen, Huu Nghia	408
Nie, Guoping	237
Noguera, Manuel	569

Nunes, João	81
Oliveira de Almeida, Hyggo	414
Oliveira Santos, Igor Peterson	466
Oliveira, Adicinéia	323
Oliveira, Elaine	595
Oliveira, Ricardo R.	396
Orucevic-Alagic, Alma	64
Paiva, Breno Lacerda A.	434
Park, Chang-Sup	18
Park, Grace	569
Park, Younghee	99
Pasquier, Nicolas	454
Patel, Hardik	659
Patel, Jvalant	386
Patrick, Matthew	382
Perkusich, Angelo	8, 81, 267, 414
Perkusich, Mirko	8, 81
Perry, Dewayne	58
Pesado, Patricia	504
Petriu, Dorina	333
Pinheiro, Placido	575
Pinheiro, Placido Rogerio	402
Pinto, Victor H. S. C.	396
Placido Da Silva, Luiz Sergio	657
Poltronieri Rodrigues, Ildevana	628
Popolin de Araújo Cunha, Marcel	231
Prates Correia, Tales	363
Precioso, Frederic	454
Pruski, Cédric	329
Qi, Hang	462
Qi, Yuanchao	655
Qin, Yong	285, 291
Qingsheng, Zhu	498
Rahman, Mohammad Masudur	521
Ramalho, Franklin	644
Ramos, Danilo	559
Ramos, Lukas T.	434
Rao, Yongsheng	141
Raza, Mushtaq	515
Reddy, Sreedhar	197
Ren, Danni	124
Ridgeway, Jefferson	486
Rivero Cabrejos, Luis Jorge Enrique	600
Rivero, Luis	595

Rolim de Sousa, Reudismam	267
Rosca, Daniela	24
Roy, Chanchal K.	426, 521
Rumpe, Bernhard	351
Saeed, Khalid	93
Sahoo, Anam	638
Salazar, Rafael	515
Sampaio, Augusto	420
Sanches Da Silva, Carlos Eduardo	611
Santos, Jadson	583
Santos, Rodrigo	317
Sant' Ana, Laís A.	555
Saraiva, Renata	8, 81
Sayeb, Khaoula	14
Selbach Silveira, Milene	628
Serai, Abdelhak	1
Shahmir, Nazlie	579
Shao, Sai	297
Shen, Beijun	87, 147, 191
Shen, Zhiqi	75
Shi, Kai	225
Shi, Lin	165
Sillitti, Alberto	249
Silva, Andreia	575
Silva, Jislane	559
Silva, Williamson	595
Simão Filho, Marum	402
Soares, Michel	323, 559
Song, Chenxi	71
Souza, Mauricio Ronny	657
Souza, Paulo S. L.	396
Souza, Rogéria C. G.	555
Souza, Simone R. S.	396
Sun, Guozi	124
Sun, Haiying	547
Sun, Jing	443
Sun, Xuan	110
Suonsyrjä, Sampo	114
Svajlenko, Jeffrey	426
Systä, Kari	114
Tadikonda, Veera	24
Tang, Xiaojun	207
Tao, Chuanqi	375, 480
Tao, Wan	203
Terho, Henri	114
Tian, Yiqiuzi	213

Tiejian, Luo	31
Vale, Sushant	197
Valêncio, Carlos R.	555
Vilkomir, Sergiy	474
Vora, Jainikkumar	386
Walia, Gursimran	531
Wan, Hongyan	171
Wang, Jiayue	37
Wang, Junjie	165
Wang, Lu	654
Wang, Qi	655
Wang, Qing	165
Wang, Tao	71
Wang, Xinzhi	37
Wang, Yabin	243
Wang, Yuanyang	547
Wang, Zhao	46
Wang, Zhongjie	58
Wang, Ziyuan	655
Willamy, Renan	81
Woodside, Murray	333
Wortmann, Andreas	351
Wu, Guoqing	171
Wu, Hong	165
Wu, Jianping	279
Xie, Zhengyu	291
Xin, Chen	31
Xu, Chiya	655
Xu, Dianxiang	492
Xu, Haiping	449
Xu, Lingyu	42
Xu, Ming	279
Xu, Xiaofei	58
Xu, Zheng	46, 52, 207
Xu, Zhengjie	243
Xu, Zhou	213
Xue, Yufeng	309
Xue, Yunlan	42
Yan, Meng	134
Yang, Cheng	71
Yang, Guowei	486
Yang, Mengning	134
Yang, Yanfang	285
Yang, Yitao	124

Yang, Yu	87
Ye, Jun	46
Ye, Wei	110
Yeddula, Raghavendra Reddy	197
Yin, Gang	71
You, Guoan	159, 171
Yu, Huiqun	225
Yu, Jie	42
Yu, Wei	203
Yu, Xiao	237
Yu, Xinjia	75
Yuan, Feng	165
Yuan, Mengting	171
Yumi Nakagawa, Elisa	363
Zafalon, Geraldo F. D.	555
Zagade, Pramod	197
Zaidi, Fatiha	408
Zhang, Dongmei	390
Zhang, Hui	37
Zhang, Li	177
Zhang, Shunxiang	207
Zhang, Weiqiang	243
Zhang, Xiaohong	134
Zhang, Xunhui	71
Zhang, Yuhan	449
Zhao, Mengjia	261, 634
Zheng, Jiabin	255
Zhou, Peng	261, 634
Zhou, Xianlin	46
Zhou, Yang	37
Zhu, Guangli	207
Zhu, Jiangang	147, 191

Program Committee Reviewers' Index

Silvia T. Acuña	Universidad Autonoma de Madrid
Oum-El-Kheir Aktouf	Grenoble INP
Taisira Al-Belushi	Sultan Qaboos University
Shadi Alawneh	C-CORE
Mark Allison	Florida International University
Izzat Alsmadi	Yarmouk University
Min An	University of Birmingham
John Anvik	University of Lethbridge
Doo-Hwan Bae	KAIST
Ebrahim Bagheri	Ryerson University
Hamid Bagheri	University of California, Irvine
Xiaoying Bai	Tsinghua University
Ellen Barbosa	ICMC/USP
Fevzi Belli	Univ. Paderborn
Ateet Bhalla	Independent Consultant
Alessandro Bianchi	Department of Informatics - University of Bari
Ivo Bukovsky	Department of Instrumentation and Control Engineering, Faculty of Mechanical Engineering, Czech Technical University in Prague
Keith C.C. Chan	The Hong Kong Polytechnic University
Chih-Hung Chang	Dept. Of Information Management, Hsiuping University of Science and Technology
Shikuo Chang	University of Pittsburgh
Wen-Hui Chen	Taipei University of Technology
Stelvio Cimato	Dipartimento di Informatica, Università degli Studi di Milano
Nelly Condori-Fernández	VU University of Amsterdam
Fabio Costa	Federal University of Goias
Maria Francesca Costabile	Dipartimento di Informatica - University of Bari
Laurent D'Orazio	UBP
Guglielmo De Angelis	ISTI-CNR
Jose Luis de La Vara	Carlos III University of Madrid
Junhua Ding	East Carolina University
Derek Doran	Wright State University
Huan Du	Shanghai University
Christof Ebert	Vector
Ali Ebneenasir	Michigan Technological University
Omar El Ariss	The Pennsylvania State University, Harrisburg
Mahmoud Elish	King Fahd University of Petroleum & Minerals
Ruby Elkharboutly	Quinnipiac University
Davide Falessi	Cal Poly, USA
Behrouz Far	University of Calgary
Liana Fong	IBM T. J. Watson Research
Jerry Gao	San Jose State University
Junwei Gao	Qingdao University
Kehan Gao	Eastern Connecticut State University
Kun Gao	Zhejiang University

Zeyu Gao	San Jose State University
Felix Garcia	Alarcos Research Group, Information and Systems Department, Escuela Superior de informática, University of Castilla-La Mancha
Ignacio García	University of Castilla-La Mancha
Raúl García-Castro	Universidad Politecnica de Madrid
Swapna Gokhale	Univ. of Connecticut
Wolfgang Golubski	Westfälische Hochschule Zwickau
Anurag Goswami	North Dakota State University
Des Greer	Queens University Belfast
Christiane Gresse von Wangenheim	Federal University of Santa Catarina - UFSC
Katarina Grolinger	UWO
Hassan Haghighi	Shahid Beheshti University
Hao Han	The University of Tokyo
Xudong He	Florida International University
Robert Heinrich	Karlsruher Institute of Technology
Rubing Huang	Jiangsu University
Shihong Huang	Florida Atlantic University
Hamdy Ibrahim	Ph.D. student
Bassey Isong	University of Venda
Clinton Jeffery	University of Idaho
Yue Jiang	Fujian Normal University
Jason Jung	Chung-Ang University
Selim Kalayci	East Tennessee State University
Ananya Kanjilal	Department of CSE, B.P.Poddar Institute of Management & Technology, Kolkata-52
Taghi Khoshgoftaar	Florida Atlantic University
Jun Kong	North Dakota State University
Aneesh Krishna	Curtin University, Australia
Vinay Kulkarni	Tata Consultancy Services
Cyril Labbé	Grenoble University
Yu Lei	University of Texas at Arlington
Bixin Li	SOUTHEAST UNIVERSITY
Yuan-Fang Li	Monash University
Zhi Li	College of Computer Science and Information Technology, Guangxi Normal University
Tao Liao	Shanghai University
Jianhua Lin	Eastern Connecticut State University
Lan Lin	Department of Computer Science, Ball State University
Hongchao Liu	
Kaikai Liu	San Jose State University
Ting Liu	Xi'an Jiaotong University
Xiaodong Liu	Edinburgh Napier University
Xingwang Liu	Southeast University
Luanna Lopes Lobato	Universidade Federal de Pernambuco - UFPE
Xiangfeng Luo	Computer department, Shanghai University, 149, Yanchang Road, Shanghai, 200072, China P.R.

Baojun Ma	Beijing University of Posts and Telecommunications
Ivan Machado	UFBA - Federal University of Bahia
Marcelo Maia	UFU
Riccardo Martoglia	FIM - University of Modena
Beatriz Marín	Universidad Diego Portales
Santiago Matalonga	Universidad ORT Uruguay
Hong Mei	Peking University
Hsing Mei	Fu Jen Catholic University
Andre Menolli	Universidade Estadual do Norte do Paraná - UENP
Ali Mili	NJIT
Alok Mishra	Atilim University, Incek 06836, Ankara - Turkey
Hiroyuki Nakagawa	Osaka University
Edson Oliveirajr	State University of Maringá
Xin Peng	Fudan University
Oscar Pereira	University of Aveiro
Angelo Perkusich	Federal University of Campina Grande - UFCG
Antonio Piccinno	University of Bari
Alfonso Pierantonio	University of L'Aquila
Daniel Plante	Stetson University
Mukul Prasad	Fujitsu Laboratories of America
Yong Qin	Beijing JiaoTung University
Rick Rabiser	Christian Doppler Lab. MEVSS, Johannes Kepler University Linz
Filip Radulovic	Ontology Engineering Group, Universidad Politécnica de Madrid
Damith Rajapakse	National University of Singapore
Rajeev Raje	IUPUI
Henrique Rebêlo	Federal University of Pernambuco - UFPE
Elder Rodrigues	PUCRS
Daniel Rodriguez	The University of Alcalá
Claudia Roncancio	Grenoble University
Samira Sadaoui	University of Regina
Seyed Masoud Sadjadi	Florida International University
Claudio Sant'Anna	Federal University of Bahia
Abdelhak Seriai	Lirmm/université Montpellier 2
Michael Shin	Texas Tech University
Martin Solari	Universidad ORT Uruguay
Qinbao Song	Xi'an Jiaotong University
George Spanoudakis	Department of Computer Science, City University
Rajesh Subramanyan	siemens
Vijayan Sugumaran	School of Business Administration, Oakland University, Rochester, MI 48309, USA
Jing Sun	The University of Auckland
Yanchun Sun	School of Electronics Engineering and Computer Science, Peking University, China
Gerson Sunyé	Université de Nantes
Chuanqi Tao	Cisco System
Mark Trakhtenbrot	Holon Institute of Technology

Peter Tröger	TU Chemnitz
T.H. Tse	The University of Hong Kong
Burak Turhan	University of Oulu
Christelle Urtado	LGI2P - Ecole des Mines d'Alès
Sylvain Vauttier	LG2IP / Ecole des Mines d'Alès
Silvia Vergilio	UFPR
Gennaro Vessio	University of Bari
Sergiy Vilkomir	East Carolina University
Arndt Vonstaa	PUC-Rio
Huanjing Wang	Western Kentucky University
Jiacun Wang	Department of Software Engineering, Monmouth University, West Long Branch, NJ 07764, USA
Xiaoyin Wang	University of Texas at San Antonio
Ye Wang	Zhejiang Gongshang University
Yong Wang	Texas A&M University
Young Wang	Chongqing University of Posts and Telecommunications
Ziyuan Wang	Nanjing University of Posts and Telecommunications
Hironori Washizaki	Waseda University
Dali Wei	University of California, Berkeley
Xiao Wei	Shanghai University
Victor Winter	University of Nebraska at Omaha
Guido Wirtz	University of Bamberg
Franz Wotawa	Technische Universitaet Graz
Zongyi Xing	Nanjing University of Technology and Engineering
Dianxiang Xu	Boise State University
Frank Xu	Bowie State University
Haiping Xu	University of Massachusetts Dartmouth
Weifeng Xu	Bowie State University
Zheng Xu	Tsinghua University, Beijing
Zhiguo Yan	Fudan University
Guowei Yang	Texas State University
Neil Yen	Aizu University
Xiaobo Yin	Anhui University of Science and Technology, Huainan, China
Huiqun Yu	East China University of Science and Technology
Du Zhang	California State University
Pengcheng Zhang	Hohei University
Shunxiang Zhang	Anhui University of Science & Technology
Tao Zhang	Northwest Polytechnical University (China)
Yong Zhang	Web and Software Technology R&D center, Tsinghua University, Beijing, P.R.China
Zhenyu Zhang	State Key Laboratory of Computer Science, Institute of Software, Chinese Academy of Sciences
Huijuan Zhou	North China University of Technology
Guangli Zhu	aust
Xingquan Zhu	Florida Atlantic University

External Reviewers' Index

Abdelwahab, Amira
Alazzam, Iyad
Aleroud, Ahmed
Assunção, Wesley K. G.
Ayora, Clara

Baek, Youngmin
Barat, Souvik
Bera, Marcio
Bernardino, Maicon
Borges, Vanessa

Calleja, Pablo
Campos, Eduardo
Cassano, Fabio
Castro Llanos, John Wilmar
Courbis, Anne-Lise

Desolda, Giuseppe
Dietz, Linus W.
Du, Huan
Duan, Feng

Feng, Huadong
Fioravanti, Maria Lydia
Freitas Duarte Filho, Nemesio

Gamage, Dimuthu
Garijo, Daniel
Geiger, Matthias
Guo, Danhuai

He, Tiantian

Jakobs, Christine
Jee, Eunkyong
Jiang, Tao

Kolb, Stefan
Krotsiani, Maria

Lambolais, Thomas
Lanzilotti, Rosa
Li, Zhongwei

Lima, Júnio
Liu, Liang
Liu, Yuzhen

Macchi, Darío
Magües, Daniel A.
Malhotra, Ruchika
Marcolino, Anderson
Martins de Andrade, Paulo
Mokni, Abderrahman
Moraga, Ma Ángeles
Mottu, Jean-Marie

Nebut, Clémentine
Niu, Xintao

Ogale, Pushkar

Paixão, Klérisson
Park, Jihun
Perkusich, Mirko
Prusa, Joey

Roberts, Michele
Rocca, Ignacio
Rostami, Kiana

Shin, Donghwan
Sobreira, Victor
Song, Jiyoung
Strittmatter, Misha
Sunkle, Sagar

Taspolatoglu, Emre
Theis Geraldi, Ricardo
Tibermacine, Chouki

Wang, Bin
Wang, Botao
Wei, Xiao
Weiss, Karl
Wu, Wei

Xu, Guangquan

Zhang, Degan
Zhang, Donghong
Zhang, Jun

Zhang, Liangbin
Zhang, Long
Zhang, Shunxiang
Zhaoxia, Yin
Zhou, Guangyu
Zhou, Pei-Yuan



Proceedings of the
Twenty-Eighth
International
Conference on
Software Engineering &
Knowledge Engineering

SEKE

San Francisco Bay July 1-3 **2016**

Copyright © 2016
Printed by
KSI Research Inc. &
Knowledge Systems Institute
Graduate School
156 Park Square
Pittsburgh, PA 15238 USA
Tel: +1-412-606-5022
Fax: +1-847-679-3166
Email: seke@ksiresearch.org
Web: <http://ksiresearchorg.ipage.com/seke/seke16.html>
Printed in USA, 2016
ISBN 1-891706-39-X (paper)
ISSN 2325-9000 (print)

