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 Departament - DSI
UFS – Federal University of Sergipe
Itabaiana/SE - Brasil.
andreviniciusnascimento@gmail.com, {mjrse, wenderson_se}@hotmail.com

***Abstract*— This paper presents an approach to automate the selection and execution of previously indentified 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.

DOI reference number: 10.18293/SEKE2016-182

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)** $\mathbf{H_{0time}}$ - the execution of automated and manual testing has same efficiency. ($\mu_{ManualTestingTime} = \mu_{AutomationTestingTime}$); **2)** $\mathbf{H_{1time}}$ - 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 3[1] 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

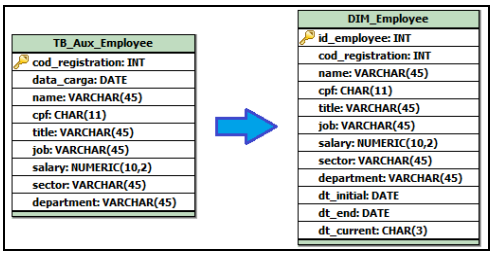

Figure 1. Charge for a dimension with behaviors of types 1 and 2.

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] Elgamal, N., Elbastawissy, A. and Galol-edeen, 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)