On the Reuse of Knowledge to Develop Intelligent Software Engineering Solutions

José Ferdinandy Silva Chagas
Federal Rural University of Semi-Arid
Intelligent Software Engineering Group
ferdinandy@ufersa.edu.br

Luiz Antonio Pereira Silva
Federal University of Campina Grande
Intelligent Software Engineering Group
luizantonio@copin.ufcg.edu.br

Mirko Perkusich
Intelligent Software Engineering Group
mirko@virtus.ufcg.edu.br

Ademar França de Sousa Neto
Federal University of Campina Grande
Intelligent Software Engineering Group
ademar.sousa@virtus.ufcg.edu.br

Danyllo Albuquerque
Intelligent Software Engineering Group
danyllo.albuquerque@virtus.ufcg.edu.br

Dalton Cézane Gomes Valadares
Federal Institute of Pernambuco
Intelligent Software Engineering Group
dalton.cezane@caruaru.ifpe.edu.br

Hyggo Almeida
Federal University of Campina Grande
Intelligent Software Engineering Group
hyggo@dsc.ufcg.edu.br

Angelo Perkusich
Federal University of Campina Grande
Intelligent Software Engineering Group
perkusic@dee.ufcg.edu.br

***Abstract*—Intelligent Software Engineering (ISE) is currently a hot topic in research. Besides being a promising field, it brings many challenges. Therefore, there is a need for guidelines to help researchers to build an ISE solution. The goal of this study is to identify patterns in developing ISE solutions. For this purpose, we analyzed 42 studies, using a thematic analysis approach, to understand how they reused knowledge and applied it to solve a SE task. As a result, we developed a thematic network composed of the main concepts related to knowledge reuse for ISE. Further, we identified that researchers use external and internal knowledge sources, and mostly rely on structured data to develop ISE solutions. Despite this, there are alternatives such as eliciting data from humans and literature to identify metrics or build knowledge-based systems. Overall, we concluded that there many research opportunities to guide the construction of ISE solutions.**

I. Introduction

The processing power of modern computers increased considerably, enabling Artificial Intelligence (AI) to reach streets, houses, cities, and people daily [22], [18]. Hence, AI advances bring new challenges and opportunities, such as automating or supporting the execution of Software Engineering (SE) tasks [27], [10]. On the other hand, bringing AI systems to the market also brings challenges that can be addressed by applying SE. As a consequence, recently, the field denominated as Intelligent Software Engineering (ISE) has emerged. ISE is an ambidextrous field focusing on (i) applying intelligent techniques to solve SE problems and (ii) using SE to improve AI systems [53], [44]. In this paper, we focus only on (i), which, itself, is not a very recent phenomenon dating back to the 1980s [37]. As the definition for what is an *intelligent technique*, we follow Perkusich et al. [44], in which this term was defined as: "the exploration of data (from digital artifacts or domain experts) for knowledge discovery, reasoning, learning, planning, natural language processing, perception or supporting decision-making".

In industry, companies such as Facebook and Amazon have been applying intelligent techniques (i.e., search-based algorithms) to solve SE problems [41]. In academia, it is a hot topic [44]. For instance, researchers have proposed the application of NLP to manage requirements [28] and the use of natural-language-based chatbots to improve the productivity of developers [25]. Moreover, through ML, the researchers can use accessible software repositories, with a lot of available data, to continuously learn and improve the software reuse [50]. Perkusich et al. [44] performed a systematic literature review on ISE in the context of agile software development and highlighted the following research themes: Search-Based Software Engineering (SBSE) [19], machine learning for SE [55], recommender systems for SE [15], Bayesian networks for SE [35], software analytics [34], Big Code [2] and decision analysis for SE [56]. Among the SE problems that researchers are addressing with intelligent techniques, we can list: Effort Estimation ([48], [51], [13], [23]), Risk Management ([9]), Software Testing ([29], [24], [42], [31]), Team Formation ([30], [7]), and Requirements Engineering ([47], [26], [38], [46]).

Developing an ISE solution is a complex task because it demands knowledge regarding the SE task at hand and intelligent techniques, and, to the best of our knowledge, there are few proposals of conceptual models or general guidelines to develop ISE solutions. The literature presents guidelines for ISE subfields such as data mining for software engineering [54], [17], machine learning for software engineering [33], Search-Based Software Engineering [20], and data-driven solutions for agile projects [10]. It also presents guidelines for applying intelligent techniques for general purposes, such as building Bayesian networks [39], [32]. Despite having their value,

DOI reference number: 10.18293/SEKE2020-157

the existing studies focus on a specific intelligent technique. The problem is that defining the intelligent technique should not be the starting point of defining an ISE solution. The solution designer only selects the intelligent technique to be applied after evaluating the existing available knowledge (e.g., data stored in CASE tools or repositories) and the software engineering problem to be tackled (e.g., estimate effort for a given task). Therefore, to help in the early stages of building an ISE solution, we argue that there is a need for general guidelines.

To address this need, we analyzed 42 studies, identified by Perkusich et al. [44], that applied intelligent techniques to several SE tasks to identify patterns and provide a holistic view on how to develop ISE solutions from the perspectives of Knowledge Management (KM) and reuse-driven software engineering.

This paper synthesizes our findings by presenting a thematic network and the identified patterns on how the applied intelligent techniques relates to the reused knowledge. Further, it discusses the implications for research and practice. The rest of the paper is structured as follows. Section II presents the applied methodology to perform the thematic analysis. Section III describes the conceptual model. Section IV discusses the model development challenges, application, and impact. Section V lists main threats to validity. Finally, Section VI presents our final remarks, emphasizing the research contribution and limitations, and suggesting future works.

II. Research Methodology

The goal of this study is to identify patterns in developing ISE solutions. For this purpose, we model the problem of developing ISE solutions from the perspective of knowledge-reuse, in which we assume that an intelligent technique reuses data, information, or knowledge, which might be available through digital artifacts or domain experts, to solve SE problems. Given this, we defined the following research questions:

- RQ1 - How is knowledge reused in the context of ISE?
- RQ2 - What is the relationship between the type of reused knowledge and the applied intelligent technique?

RQ1 focuses on classifying existing ISE solutions in terms of the type of knowledge sources used and, if the case, what are the knowledge transformation techniques employed by researchers to feed intelligent technique algorithms.

RQ2 focuses on identifying patterns between the type of reused knowledge and the applied intelligent techniques. The answer to this research question might indicate trends, which might serve as guide researchers and practitioners interested in developing ISE solutions.

To answer the research questions, we employed a thematic analysis approach following the guideline proposed by Cruzes and Dyba [8]. The guideline proposes five research steps: (i) data extraction, (ii) code data, (iii) translate codes into themes, (iv) create a model of higher-order themes, and (v) assess the trustworthiness of the synthesis.

As the data source for the first step (i), we used a subset of the studies reported by Perkusich et al. [44], which identified intelligent techniques applied to agile software development. Despite restricting the scope to agile, Perkusich et al. [44] report ISE solutions for diverse SE tasks such as effort estimation, requirements prioritization, and risk management; and using digital artifacts and humans expertise as knowledge sources. Therefore, we judged that analyzing the ISE solutions reported by Perkusich et al. [44] as being sufficient, given the scope of this study.

To assure that our results are based only on high-quality studies, guaranteeing the trustworthiness of the synthesis (step v), we filtered the 104 papers following the quality scoring performed by Perkusich et al. [44]. Perkusich et al. [44] used the instrument proposed by Dyba and Dingoyr [11] to assess the quality of the studies. We present the quality criteria in what follows.

1) Is the paper based on research (or is it merely a “lessons learned” report based on expert opinion)?
2) Is there a clear statement of the aims of the research?
3) Is there an adequate description of the research context?
4) Was the research design appropriate to address the aims of the research?
5) Was the recruitment strategy appropriate to the aims of the research?
6) Was there a control group with which to compare treatments?
7) Was the data collected in a way that addressed the research issue?
8) Was the data analysis sufficiently rigorous?
9) Has the relationship between the researcher and participants been considered to an adequate degree?
10) Is there a clear statement of findings?
11) Is the study of value for research or practice?

For each quality criteria, Perkusich et al. [44] rated the studies using a boolean scale in which “1” means “yes” and “0” means “no”. Therefore, the quality score ranges in the interval [0, 11], which is composed only of Integer numbers. We only evaluated studies with a quality score equal to seven or higher, resulting in 42 studies. The complete list of evaluated study is made available here[1].

After having identified the 42 studies to be used as our data source, one researcher analyzed them and extracted publication details (e.g., title and year), the applied intelligent technique, SE task, level of automation [12] and segments of text relevant given our research questions. Afterward (step ii), each segment of text was analyzed and labeled by a researcher, generating a set of codes. We used an integrated approach, in which we defined a “start list” of codes based on our expertise in the field, but we remained open for new concepts that could become apparent. Another researcher checked the coded segments to avoid researcher bias.

Examples of codes defined a priori are the type of knowledge source (i.e., tacit, explicit, or both), and type of data (i.e., structured, non-structured, both). Conversely, as an example of a code that became apparent during data analysis was techniques to “transform” unstructured data into structured data such as text mining, ontology, and qualitative analysis. We discuss these concepts in Section III.

[1]https://bit.ly/2Q596MK

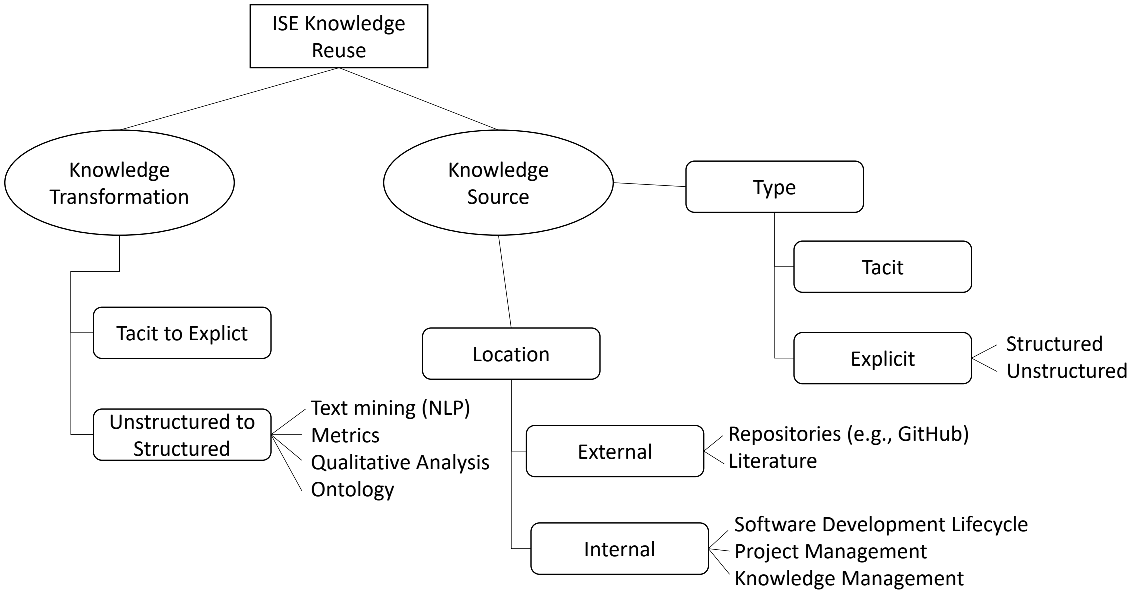

Fig. 1. ISE Knowledge Reuse thematic network

To define and structure the themes (steps iii and iv), the researchers analyzed the codes during workshops. At the end of this process, we developed a thematic network [3], organizing the concepts related to knowledge reuse for ISE. Afterward, we used the identified themes to classify the studies and analyze our research questions.

III. Thematic Network

This section discusses the thematic network, shown in Figure 1, that resulted from analyzing the 42 papers from Perkusich et al. [44], as discussed in Section II. The thematic network focuses on structuring the concepts related to knowledge reuse in the context of ISE. Therefore, the Global Theme (represented as a rectangle in Figure 1) encompasses the ISE solutions reuses knowledge.

During our analysis, two middle-order themes (represented by ellipsis in Figure 1) emerged: *Knowledge Source* and *Knowledge Transformation*. Every ISE solution, in some way or another, uses knowledge for a SE task. The *Knowledge Source* theme represents the possible types of sources in which the solution designer or algorithm might collect the necessary knowledge. We further refined this theme by identifying the themes *Type* and *Location*.

The theme *Type* refers to the two possible types of knowledge described in classical KM literature: *Tacit* or *Explicit*. Tacit knowledge refers to knowledge that is only stored in the minds of stakeholders (e.g., programmers, software engineers, and project managers). Explicit knowledge refers to knowledge that is codified and stored in digital (or physical) artifacts. There are two types of explicit knowledge, *Structured* and *Unstructured*. We defined that structured data refers to data high-organized and easily processed by a machine (e.g., relational database search). Conversely, unstructured data cannot be processed using conventional tools. In our context, mostly, unstructured data refers to text (e.g., requirements, system logs, and source code) but could include audio and video, for instance.

The *Location* theme characterizes where the necessary knowledge might be found. For this theme, we identified two options: *External* and *Internal*. An external source refers to sources that are external to a given organization, such as a repository and the literature. Many researchers in data-driven ISE use repositories such as GitHub [16] for data mining, but the literature is an important source of knowledge for ISE. For instance, Hearty et al. [21], Perkusich et al. [45] and Freire et al. [14] identified features for their proposed models, partially, based on information collected from the scientific and grey literature.

Internal sources refer to knowledge that is available within an organization. It is the case for data produced during the Software Development Lifecycle, Project Management, and Knowledge Management activities. Given this, it is vital to notice that there are cases in which the necessary data is not readily available to solve de SE task at hand. In these cases, the ISE designer must develop tools to collect such data and integrate it them existing processes followed by the organization or evaluate the possibility of transforming existing knowledge into usable data for intelligent techniques, which is discussed in what follows.

The *Knowledge Transformation* theme refers to transforming knowledge that is available, but not ready to be used for ISE. It is the case when we have unstructured or tacit knowledge that we wish to use. In the case of unstructured data, it is necessary to transform it into structured. We identified such as text mining (e.g., Natural Language Processing) [28], the use of software metrics [36], qualitative analysis (e.g., coding) [49] and ontology [6] for this purpose. In the case of having tacit knowledge, it must be transformed into explicit. This process

Theme	%	Freq.	Distribution
Intelligent Technique			
Search and optimization	21%	9	
Reasoning under uncertainty	19%	8	
Mathematical model	14%	6	
Multiple Criteria Decision Analysis	7%	3	
Machine learning	21%	9	
Rules	5%	2	
Others	12%	5	
Level of Automation			
3	45%	19	
4	55%	23	
Non-Structured to Structured			
None	10%	4	
Metrics	72%	28	
Text mining (NLP)	5%	2	
Qualitative analysis (	8%	3	
Others	13%	5	
Uses tacit knowledge?			
No	62%	26	
Yes	38%	16	
Number of experts			
0-10	89%	40	
11-50	4%	2	
> 50	7%	3	
External Type			
Repository	45%	19	
Scientific Literature	38%	16	
None	14%	6	
Internal type			
Software Development Process	31%	13	
Project Management Process	62%	26	
Knowledge Management Process	2%	1	
None	10%	4	
Data type			
Only structured	83%	35	
Both	17%	7	
Only unstructured	0 %	0	

Fig. 2. Frequencies of Themes.

might transform tacit knowledge structured or unstructured. In the latter case, it is necessary to transform it into structured. For instance, Perkusich et al. [43] elicited knowledge from 46 Scrum experts through the Delphi method and an online survey to construct a Bayesian network for assessing Scrum projects.

IV. Discussion

This section discusses the research questions (see Sections IV-A and IV-B) presented in Section II and the implications for research and practice (see Section IV-C).

A. ISE Knowledge Reuse

We used the thematic network presented in Figure 1 to classify the studies and identify the trends on knowledge reuse for ISE. Figure 2 presents the frequencies for each theme, as they were extracted from the studies. In what follows, we discuss each of the results for each of the basic themes (represented as rectangles with rounded borders in Figure 1 presented in the thematic network.

For the theme *Location*, 36 (86%) studies used some form of external knowledge source, while 42 (95%) used internal sources. Regarding the studies that relied on external sources, 19 studies used data from a repository and 16, from the scientific literature. Regarding the studies that relied on internal sources, 13 studies collected data from artifacts produced during the Software Development Lifecycle, 26 from artifacts produced by the Project Management Process, and only 1 from Knowledge Management Processes. The collected data indicates that researchers when developing ISE solutions, look for wherever places necessary to find data. Despite this, care should be taken when deploying ISE solutions in practice, because having different data sources raises the complexity in operating and maintaining them.

For the theme *Type*, 16 studies used tacit knowledge in the development cycle of the ISE solution, mostly (89%) eliciting it from ten or fewer experts. For the development cycle, we included a potential evaluation of the developed ISE solution by humans. Therefore, we considered that studies that developed expert systems such as Perkusich et al. [45], Odzaly et al. [40] and data-driven studies that evaluated their solution with humans (e.g., Chaves-González et al. [4]) equally. The reasoning applied is that either way, the tacit knowledge of humans was used to develop the ISE solution. It is essential to notice that we did consider here human knowledge regarding the intelligent technique itself (e.g., knowledge to define the fitness function of a genetic algorithm), but only related to the SE task at hand.

Further, 35 (83%) used only structured data, while 7 (17%) used both types of data (i.e., structured and unstructured). For instance, Hearty et al. [21] used only structured data (i.e., metrics and rules in the form of probability functions) to build a Bayesian network to predict the velocity of an XP team. Conversely, Lucassen et al. [28] presents a tool that uses metrics (i.e., structured data) as indicators of a well-written user story and processes user stories (i.e., unstructured data), calculating a quality score for them.

Regarding the theme *Knowledge Transformation*, we observed that the most popular form to transform tacit knowledge to explicit is in the form of questionnaires such as done by Perkusich et al. [43]. Further, regarding the transformation of unstructured to structured data, we observed that the most popular procedure is to use software metrics (28 studies). For instance, Abouelela and Benedicenti [1] define a set of metrics and use them to build a Bayesian network to predict the velocity and delivered quality of an XP team. A few studies used Qualitative Analysis (3) and Text mining (2). For instance, Lucassen et al. [28] process text from user stories using Natural Language Processing algorithms. Therefore, we identify a pattern, in the evaluated studies, to use software metrics as the preferred means for knowledge transformation.

B. Relationship between type of reused knowledge and applied intelligent technique

We triangulated the results of our classification (see Section IV-B with the data collected by Perkusich et al. [44] regarding the types of intelligent techniques applied by the studies. As presented in Figure 2, 18 studies focus on, necessarily, data-driven solutions, this is the case for *Search and Optimization* and *Machine learning*. The remaining intelligent techniques might be applied as a result of eliciting expert knowledge or exploring digital artifacts. For instance, it is the case for Bayesian networks, which can be constructed solely based on expert knowledge, available data, or both. Therefore, as expected, if there is enough structured data, any of the intelligent techniques presented by Perkusich et al. [44] might be used. Unfortunately, in practice, most of the artifacts produced by the software development process are unstructured; the case for requirements, test cases, source code, and system logs. In these cases, researchers have used tools to process unstructured data extracting metrics (e.g., Chidamber and Kemerer metrics [5]) from it automatically or processing them using text mining algorithms.

In the cases of not having enough data, there are two alternatives: (i) elicit data from humans or literature to identify metrics or (ii) build knowledge-based systems. For the first case, after identifying the metrics that are crucial to solve the SE task at hand, it is necessary to develop tools to, ideally, collect them automatically during software development or project management activities. Given that the necessary data is available, a data-driven intelligent technology such as Machine Learning can be used to infer new knowledge or make predictions, for instance. The second option is to extract knowledge from experts and develop, for instance, an expert system using a Bayesian network or a Rules-based system (e.g., production rules). In this case, ideally, it is necessary to develop a tool that collects the input automatically or from humans to infer whatever is necessary (e.g., estimate effort for a given task).

C. Implications for research and practice

This study has several implications for research and practice. For research, we have mapped how knowledge is used for ISE solutions and identified patterns on how the type of reused knowledge relates to the applied intelligent technique. The reported information might guide researchers to develop ISE solutions having a more holistic view of their development process. Despite this, the analyzed studies focus on supporting decision-making, not having a high level of automation, using the classification described in Feldt et al. [12]. Therefore, we believe that there is a need to further refine the presented thematic network by analyzing studies with higher levels of automation. Further, there is a need for more studies to define guidelines for researchers in building ISE solutions, through the form of checklists, catalogs, taxonomies or reference models; especially, focusing on the early stages of developing an ISE solution, which, usually, relies on evaluating the available knowledge to solve the SE task at hand.

Also, this study showed that researchers, when building ISE solutions, rely on publicly available repositories. The implications for this point are twofold: first, it demonstrates the importance of having data available to build ISE solutions, and second, to be able to validate and compare ISE solutions focusing on the same SE task.

For practitioners, this study shows how knowledge is reused by ISE solutions and can be an inspiration for them to use tools and participate in research that helps to make explicit knowledge available. As a consequence, better ISE tools can be developed that can, potentially, make them more efficient.

V. Threats to validity

This section discusses this study's threats to validity following the classification proposed by Wohlin et al. [52]: construct, internal, conclusion, and external validity.

- *Construct validity*: we analyzed the studies following a thematic analysis approach, in which multiple researchers participated to avoid bias. Despite this, it is possible that the resulting thematic network (Figure 1) and extracted data (Figure 2) are not representative of the are due to subjective bias.
- *Internal validity*: to assure credibility in our findings, multiple researchers checked the extracted coding, themes, and the data presented in Figure 2.
- *Conclusion validity*: since we classified the study to identify patterns using the developed thematic network, there is the risk that, since there is a threat to the construct validity of the thematic network, it influenced the extracted data and, consequently, our conclusions regarding the relationship between concepts.
- *External validity*: Moreover, since the analyzed studies focus on supporting decision-making, they do not represent all types of ISE solutions. Therefore, the constructed thematic network might not be representative of ISE solutions with higher levels of automation. Despite this, we believe that including ISE solutions with higher levels of automation might identify more basic terms for high granular, but would not have impact middle-order themes, since they follow from classic KM concepts.

VI. Conclusion

In this study, we explored patterns in developing ISE solutions, focusing on knowledge reuse by analyzing 42 papers. As a result, we developed a thematic network that relates the main concepts in this topic. Further, we identified that researchers use external and internal knowledge sources, and mostly rely on structured data to develop ISE solutions. Despite this, we showed alternatives, such as eliciting data from humans and literature to identify metrics and build knowledge-based systems (e.g., expert systems) when structured data is not readily available to be used for solving a SE task.

The main limitation of the study is only having evaluated ISE solutions that focus on supporting decision-making. Further, the study also identifies several opportunities for future work, including refining the thematic network by analyzing ISE solutions with higher levels of automation and defining guidelines for researchers to build ISE solutions, especially, giving instructions on the early-stages process of an ISE solution conceptualization.

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] M. Allamanis, E. T. Barr, P. Devanbu, and C. Sutton. A survey of machine learning for big code and naturalness. *ACM Computing Surveys (CSUR)*, 51(4):81, 2018.

[3] J. Attride-Stirling. Thematic networks: an analytic tool for qualitative research. *Qualitative research*, 1(3):385–405, 2001.

[4] J. M. Chaves-González, M. A. Pérez-Toledano, and A. Navasa. Software requirement optimization using a multiobjective swarm intelligence evolutionary algorithm. *Knowledge-Based Systems*, 83(Supplement C):105 – 115, 2015.

[5] S. R. Chidamber and C. F. Kemerer. A metrics suite for object oriented design. *Software Engineering, IEEE Transactions on*, 20(6):476–493, 1994.

[6] R. Colomo-Palacios, I. González-Carrasco, J. L. López-Cuadrado, and Á. García-Crespo. Resyster: A hybrid recommender system for scrum team roles based on fuzzy and rough sets. *International Journal of Applied Mathematics and Computer Science*, 22(4):801–816, 2012.

[7] A. A. M. Costa, F. B. A. Ramos, M. Perkusich, A. S. Freire, H. O. Almeida, and A. Perkusich. A search-based software engineering approach to support multiple team formation for scrum projects. In *SEKE*, pages 474–473, 2018.

[8] D. S. Cruzes and T. Dyba. Recommended steps for thematic synthesis in software engineering. In *2011 international symposium on empirical software engineering and measurement*, pages 275–284. IEEE, 2011.

[9] H. K. Dam. Artificial intelligence for software engineering. *XRDS: Crossroads, The ACM Magazine for Students*, 25(3):34–37, 2019.

[10] H. K. Dam, T. Tran, J. Grundy, A. Ghose, and Y. Kamei. Towards effective ai-powered agile project management. In *2019 IEEE/ACM 41st International Conference on Software Engineering: New Ideas and Emerging Results (ICSE-NIER)*, pages 41–44. IEEE, 2019.

[11] T. Dybå and T. Dingsøyr. Empirical studies of agile software development: A systematic review. *Information and software technology*, 50(9-10):833–859, 2008.

[12] R. Feldt, F. G. de Oliveira Neto, and R. Torkar. Ways of applying artificial intelligence in software engineering. In *2018 IEEE/ACM 6th International Workshop on Realizing Artificial Intelligence Synergies in Software Engineering (RAISE)*, pages 35–41. IEEE, 2018.

[13] G. R. Finnie and G. E. Wittig. Ai tools for software development effort estimation. In *Proceedings 1996 International Conference Software Engineering: Education and Practice*, pages 346–353. IEEE, 1996.

[14] A. Freire, M. Perkusich, R. Saraiva, H. Almeida, and A. Perkusich. A bayesian networks-based approach to assess and improve the teamwork quality of agile teams. *Information and Software Technology*, 100:119–132, 2018.

[15] M. Gasparic and A. Janes. What recommendation systems for software engineering recommend: A systematic literature review. *Journal of Systems and Software*, 113(Supplement C):101 – 113, 2016.

[16] G. Gousios and D. Spinellis. Mining software engineering data from github. In *2017 IEEE/ACM 39th International Conference on Software Engineering Companion (ICSE-C)*, pages 501–502. IEEE, 2017.

[17] M. Halkidi, D. Spinellis, G. Tsatsaronis, and M. Vazirgiannis. Data mining in software engineering. *Intelligent Data Analysis*, 15(3):413–441, 2011.

[18] M. Harman. The role of artificial intelligence in software engineering. In *2012 First International Workshop on RAISE*, pages 1–6. IEEE, 2012.

[19] M. Harman, S. A. Mansouri, and Y. Zhang. Search-based software engineering: Trends, techniques and applications. *ACM Comput. Surv.*, 45(1):11:1–11:61, Dec. 2012.

[20] M. Harman, S. A. Mansouri, and Y. Zhang. Search-based software engineering: Trends, techniques and applications. *ACM Computing Surveys (CSUR)*, 45(1):11, 2012.

[21] 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.

[22] H. Hourani, A. Hammad, and M. Lafi. The impact of artificial intelligence on software testing. In *2019 IEEE JEEIT*. IEEE, 2019.

[23] S.-J. Huang, N.-H. Chiu, and L.-W. Chen. Integration of the grey relational analysis with genetic algorithm for software effort estimation. *European Journal of Operational Research*, 188(3):898–909, 2008.

[24] H. Jin, Y. Wang, N.-W. Chen, Z.-J. Gou, and S. Wang. Artificial neural network for automatic test oracles generation. In *International Conf. on Computer Science and Software Engineering*. IEEE, 2008.

[25] C. Lebeuf, A. Zagalsky, M. Foucault, and M.-A. Storey. Defining and classifying software bots: a faceted taxonomy. In *Proceedings of the 1st Intern. Workshop on Bots in Software Engineering*. IEEE Press, 2019.

[26] J. Lin, M. S. Fox, and T. Bilgic. A requirement ontology for engineering design. *Concurrent Engineering*, 4(3):279–291, 1996.

[27] M. Lowry and M. R. Lowry. Knowledge-based software engineering. In *Handbook of Artificial Intelligence, volume IV*, pages 241–322. Addison-Wesely, 1989.

[28] G. Lucassen, F. Dalpiaz, J. M. E. van der Werf, and S. Brinkkemper. Improving agile requirements: the quality user story framework and tool. *Requirements Engineering*, 21(3):383–403, 2016.

[29] D. J. Mala, V. Mohan, and M. Kamalapriya. Automated software test optimisation framework–an artificial bee colony optimisation-based approach. *IET software*, 4(5):334–348, 2010.

[30] L. S. Marcolino, A. X. Jiang, and M. Tambe. Multi-agent team formation: diversity beats strength? In *Twenty-Third International Joint Conference on Artificial Intelligence*, 2013.

[31] A. M. Memon, M. E. Pollack, and M. L. Soffa. Hierarchical gui test case generation using automated planning. *IEEE TSE*, 2001.

[32] E. Mendes. *Practitioner's knowledge representation: a pathway to improve software effort estimation*. Springer Science & Business, 2014.

[33] T. Menzies. Practical machine learning for software engineering and knowledge engineering. In *Handbook of Software Engineering and Knowledge Engineering: Volume I: Fundamentals*, pages 837–862. World Scientific, 2001.

[34] T. Menzies and T. Zimmermann. Software analytics: so what? *IEEE Software*, 30(4):31–37, 2013.

[35] A. T. Misirli and A. B. Bener. Bayesian networks for evidence-based decision-making in software engineering. *IEEE Transactions on Software Engineering*, 40(6):533–554, June 2014.

[36] R. Moser, W. Pedrycz, and G. Succi. Incremental effort prediction models in agile development using radial basis functions. In *SEKE*, pages 519–522. Citeseer, 2007.

[37] J. Mostow. Foreword what is ai? and what does it have to do with software engineering? *IEEE Transactions on Software Engineering*, (11):1253–1256, 1985.

[38] J. Mylopoulos, L. Chung, and E. Yu. From object-oriented to goal-oriented requirements analysis. *Communications of the ACM*, 1999.

[39] M. Neil, N. Fenton, and L. Nielson. Building large-scale bayesian networks. *The Knowledge Engineering Review*, 15(3):257–284, 2000.

[40] E. E. Odzaly[1] and D. S. Des Greer[1]. Lightweight risk management in agile projects. 2014.

[41] I. Ozkaya. The golden age of software engineering [from the editor]. *IEEE Software*, (1):4–10, 2019.

[42] S. Parnami, K. Sharma, and S. V. Chande. A survey on generation of test cases and test data using artificial intelligence techniques. *International Journal of Advances in Computer Networks and its Security*, 2(1), 2012.

[43] 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, 2013.

[44] M. Perkusich, L. C. e Silva, A. Costa, F. Ramos, R. Saraiva, A. Freire, E. Dilorenzo, E. Dantas, D. Santos, K. Gorgônio, et al. Intelligent software engineering in the context of agile software development: A systematic literature review. *Information and Software Technology*, 119:106241, 2020.

[45] 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.

[46] F. B. A. Ramos, A. A. M. Costa, M. Perkusich, H. O. Almeida, and A. Perkusich. A non-functional requirements recommendation system for scrum-based projects. In *SEKE*, pages 149–148, 2018.

[47] H. B. Reubenstein and R. C. Waters. The requirements apprentice: Automated assistance for requirements acquisition. *IEEE TSE*, 1991.

[48] K. Srinivasan and D. Fisher. Machine learning approaches to estimating software development effort. *IEEE TSE*, 21(2):126–137, 1995.

[49] A. Turani. Applying case based reasoning in agile software development. *Journal of Theoretical and Applied Information Technology*, 78(1):120, 2015.

[50] D. P. Wangoo. Artificial intelligence techniques in software engineering for automated software reuse and design. In *4th ICCCA*. IEEE, 2018.

[51] J. Wen, S. Li, Z. Lin, Y. Hu, and C. Huang. Systematic literature review of machine learning based software development effort estimation models. *Information and Software Technology*, 54(1):41–59, 2012.

[52] C. Wohlin, P. Runeson, M. Höst, M. C. Ohlsson, B. Regnell, and A. Wesslén. *Experimentation in software engineering*. Springer Science & Business Media, 2012.

[53] T. Xie. Intelligent software engineering: Synergy between ai and software engineering. In *Intern. Symposium on DSE*. Springer, 2018.

[54] T. Xie, S. Thummalapenta, D. Lo, and C. Liu. Data mining for software engineering. *Computer*, 42(8):55–62, 2009.

[55] D. Zhang and J. J. Tsai. Machine learning and software engineering. *Software Quality Journal*, 11(2):87–119, Jun 2003.

[56] L. Zhu, A. Aurum, I. Gorton, and R. Jeffery. Tradeoff and sensitivity analysis in software architecture evaluation using analytic hierarchy process. *Software Quality Journal*, 13(4):357–375, 2005.