FOUNDATIONS OF COMPUTING AND DECISION SCIENCES
Vol. 43 (2018) No. 4
ISSN 0867-6356
DOI: 10.1515/fcds-2018-0013 e-ISSN 2300-3405

sciendo

Introduction to the Special Issue on Software Engineering Methods, Tools and Products Improvement and Evaluation

Lech Madeyski * Miroslaw Ochodek †

1. Introduction

Software Engineering plays an increasingly critical role in our daily lives, whether the developed software is in the computer applications we use in a wide range of devices (e.g., PCs, tablets, smartphones), the vehicles we drive, the air planes we fly in, or the medical equipment we require (e.g., pacemakers). The increasing inherent complexity of software systems and market competitiveness pose real challenges that need to be addressed by the Software Engineering processes, methods and tools, while the proposed methods, related artefacts and tools, as well as delivered products need to be evaluated. Software engineers continuously struggle to improve their software engineering methods in order to deliver high quality software products. As the software products built nowadays are bigger and bigger, bug free and quickly developed software may seem like an unattainable goal. Nevertheless, researchers and practitioners are obliged (and this is our aim) to improve and evaluate software engineering methods and delivered products. According to Gartner, the size of the worldwide software industry 5 years ago (in 2013) was over US$400 billion [6]. Hence, good recommendations how to improve software engineering processes and how to deliver high quality software are of huge (not only, but also monetary) value. The more so that problems with the software quality are widespread, while at the same time our life depends on software more and more every year.

According to MIT Technology Review [5], 48% of organizations planned to use machine learning to gain a greater competitive advantage in 2017. Among them, 44% were expecting to benefit from machine learning by improving the efficiency of internal processes (operations). With the advancements that have been made in the area of data analytics and machine learning, we observe a growing importance of *information* as an important component when solving many Software-Engineering-related

*Wroclaw University of Science and Technology, Faculty of Computer Science and Management, Wybrzeże Wyspiańskiego 27, Wrocław, Poland, Lech.Madeyski@pwr.edu.pl
†Poznan University of Technology, Faculty of Computing, ul. Piotrowo 2, Poznań 60-965, Poland, Miroslaw.Ochodek@cs.put.poznan.pl

problems. The papers presented in this Special Issue follow this trend and focus on applying data analytics/measurement, machine learning, and knowledge engineering techniques to support software development.

This Special Issue contains six papers. The first two contributions regard using measurement to support decision-making in software projects. The paper by Kamulegeya et al. ("*Measurements in the Early Stage Software Start-ups: A Multiple Case Study in a Nascent Ecosystem*") aims at filling the existing gap [10] in understanding how measurement is performed in software start-ups in East Africa. The results of a multi-case study involving 19 software start-up projects show that there is a visible difference in the types of measures used in the start-up projects in Africa and in their counterparts from more developed countries. The authors observed that the early-stage software start-ups in the nascent East Africa ecosystem prefer to use business and product metrics and to a less extent organizational performance metrics.

Continuous integration has gained a lot of attention in recent years [8]. In the second paper ("*Measurement and Impact Factors of Speed of Reviews and Integration in Continuous Software Engineering*"), Staron et al. aims at understanding which factors influence the duration of code reviews and software-product integration in companies employing continuous software development. After conducting a case study in two large companies developing embedded software, they identified 19 factors that affected the duration of review and integration. Although most of these factors were specific for the studied companies, being aware of them could help other companies to reduce the duration of code reviews and integration without compromising the quality of their products.

The second group of three papers regards the applying prediction models to support software development activities. The paper by Vetrò et al. ("*Combining Data Analytics with Team Feedback to Improve the Estimation Process in Agile Software Development*") concerns effort estimation in agile software development projects. In agile projects, development effort is usually estimated using expert judgment or group estimation methods [3, 11] and often at the level of a single requirement, user story, or task since applying algorithmic estimation models at the level of a single task could be difficult due to a large variability and number of factors (often unknown) that affect the actual effort [4]. The authors of the paper, propose to combine data analytics techniques with project team feedback to improve the estimation process based on user stories and story points. In particular, they propose to use a shorter non-numerical scale to express story points, an analogy-based estimation process, and retrospective analyses on the accuracy of previous sprints estimates. They report a major improvement in estimation accuracy (up to 45%) when the proposed approach was implemented in a medium-size software development company located in Germany.

The following paper by Radliński ("*Predicting Aggregated User Satisfaction In Software Projects*") aims at predicting user satisfaction in software development projects. The author applies and evaluates the accuracy of different machine learning algorithms to predict user satisfaction in the data from an extended ISBSG dataset [1]. After building, evaluating and comparing a total of 15,600 prediction schemes the best prediction scheme were selected that allowed obtaining the prediction quality mea-

sured with Matthews Correlation Coefficient (MCC), as recommended by Shepperd et al. [9], at the level of 0.6, which is a promising result for an early study.

Software maintainability is understood as the ease with which the existing software can be modified. Over the years, numerous models to measure and predict maintainability at different stages of the software development life cycle were proposed. In the fifth paper presented in this Special Issue ("*Comparative Analysis of Oriented Object Software Maintainability Prediction Models*"), Zighed et al. share the results of a comparative analysis of Object-Oriented software maintainability prediction models and propose a detailed classification of such models. The authors compare the models from three perspectives: architecture, design, and code levels, and discuss their strengths and weaknesses.

Last, but not least, the paper "*Universal Framework For OWL2 Ontology Transformations*" by Hnatkowska and Wroniecki addresses the problem of transforming domain ontologies to software code. The authors propose a framework for universal ontology processing, dedicated to ontologies expressed in OWL 2 [2] and show a proof-of-concept implementation capable of transforming OWL 2 to Groovy, which complements efforts of transformation of UML diagrams to their OWL 2 representation [7]. The proposed framework is based on a component architecture with well-defined interfaces used for communication, which makes it highly extensible.

The papers presented in this Special Issue follow the currently observed trend on using information to support product development. They show different applications of vastly understood data analytics, machine learning, and knowledge engineering to support software development processes. We believe that the presented results will inspire other researchers and allow practitioners to improve the software development processes in their organizations.

Acknowledgment

We would like to thank the authors and reviewers for their efforts to successfully complete this special section. Thanks to the excellence of the reviews, the authors benefited from the detailed feedback. Last but not least, we would like to thank Prof. Stefanowski, *Foundations of Computing and Decision Sciences* Editor-in-Chief, and Irmina Masłowska, Managing Editor of the journal, for continuous support in managing the review process and preparing this special issue.

References

[1] ISBSG Repository Data Release 11. International Software Benchmarking Standards Group, 2009.

[2] OWL 2 Web Ontology Language. Document Overview (Second Edition), W3C, 11 December 2012, 2009.

[3] Kassab M. An empirical study on the requirements engineering practices for agile software development. In *Software Engineering and Advanced Applications (SEAA), 2014 40th EUROMICRO Conference on*, pages 254–261. IEEE, 2014.

[4] Kowalska J. and Ochodek M. Supporting analogy-based effort estimation with the use of ontologies. *e-Informatica Software Engineering Journal*, 8(1):53–64, 2014.

[5] MIT Technology Review. Machine Learning: The New Proving Ground For Competitive Advantage. `https://www.technologyreview.com/s/603872/machine-learning-the-new-proving-ground-for-competitive-advantage/`. Accessed: 2017-09-03.

[6] Pettey C. Gartner says worldwide software market grew 4.8 percent in 2013. *Gartner, Stamford, Connecticut*, 2014.

[7] Sadowska M. and Huzar Z. Representation of uml class diagrams in owl 2 on the background of domain ontologies. *e-Informatica Software Engineering Journal*, 13(1):63–104, 2019.

[8] Shahin M., Babar M. A., and Zhu L. Continuous integration, delivery and deployment: a systematic review on approaches, tools, challenges and practices. *IEEE Access*, 5:3909–3943, 2017.

[9] Shepperd M., Bowes D., and Hall T. Researcher Bias: The Use of Machine Learning in Software Defect Prediction. *IEEE Transactions in Software Engineering*, 40(6):603–616, 2014.

[10] Unterkalmsteiner M., Abrahamsson P., Wang X., Nguyen-Duc A., Shah S., Bajwa S. S., Baltes G. H., Conboy K., Cullina E., Dennehy D., Edison H., Fernandez-Sanchez C., Garbajosa J., Gorschek T., Klotins E., Hokkanen L., Kon F., Lunesu I., Marchesi M., Morgan L., Oivo M., Selig C., Seppänen P., Sweetman R., Tyrväinen P., Ungerer C., and Yagüe A. Software startups – a research agenda. *e-Informatica Software Engineering Journal*, 10(1):89–124, 2016.

[11] Usman M., Mendes E., and Börstler J. Effort estimation in agile software development: A survey on the state of the practice. In *Proceedings of the 19th International Conference on Evaluation and Assessment in Software Engineering*, page 12. ACM, 2015.