Complex Systems Informatics and Modeling Quarterly (CSIMQ)
eISSN: 2255-9922
Published online by RTU Press, https://csimq-journals.rtu.lv
Article 177, Issue 32, September/October 2022, Pages I–II
https://doi.org/10.7250/csimq.2022-32.00

Handling Complexity in Modern Software Engineering: Editorial Introduction to Issue 32 of CSIMQ

Alfred Zimmermann[1] and Bartosz Marcinkowski [2]*

[1] Herman Hollerith Zentrum, Reutlingen University,
Danziger Str. 6, 71034 Böblingen, Germany
[2] Department of Business Informatics, University of Gdansk,
Piaskowa 9, 81-864 Sopot, Poland

alfred.zimmermann@reutlingen-university.de,
bartosz.marcinkowski@ug.edu.pl

This issue of CSIMQ presents three recent articles on modern software engineering. First, we focus on continuous software development and place it in the context of software architectures [1] and digital transformation [2]. The first contribution is followed by the description of the basis of specific security requirements and adequate digital monitoring mechanisms. Finally, we present a practical example of digital management of livestock farming.

Current companies and organizations are adapting their strategy, business models, products, and services as well as business processes and information systems in order to increase their level of digitalization [3], [2] through smart services and digitally enhanced products. The potential of the Internet and related digital technologies, such as the Internet of Things (IoT), cognition and artificial intelligence, data analytics, services computing, cloud computing, mobile systems, collaboration networks, and cyber-physical systems, are both strategic drivers and enablers of modern digital platforms with fast-evolving ecosystems of intelligent services for digital products.

Digitalization fosters the development of IT solutions with many globally available and diverse, rather small and distributed structures – like the aforementioned IoT or mobile systems. This has a substantial impact on architecting intelligent digital services and products while bottom-up integrating highly distributed intelligent systems. Data, information, and knowledge are fundamental core concepts of our everyday activities and are driving the digital transformation of today's global society. New services and smart connected products expand physical components by adding information and connectivity services using the Internet.

Based on an extended multiple-stage reviewing process the journal accepted the following articles:

- Theunissen, Hoppenbrouwers, and Overbeek developed novel approaches for documentation in continuous software development by writing and reading about software products to a bare minimum. In this contribution, authors distinguish between the knowledge required upfront to start a project or iteration, the knowledge required to complete a project

* Corresponding author

Reference: A. Zimmermann and B. Marcinkowski, "Handling Complexity in Modern Software Engineering: Editorial Introduction to Issue 32 of CSIMQ," Complex Systems Informatics and Modeling Quarterly, CSIMQ, no. 32, pp. I–II, 2022. Available: https://doi.org/10.7250/csimq.2022-32.00

Additional information. Author ORCID iD: A. Zimmermann – https://orcid.org/0000-0003-3352-7207 and B. Marcinkowski – https://orcid.org/0000-0002-7230-2978. PII S225599222200177X. Received: 28 October 2022. Available online: 28 October 2022.

or iteration, and knowledge required to operate and maintain software products. The authors have proposed three novel approaches to keep up with modern development methods to prevent the risk of knowledge vaporization in software projects. These novel approaches [4] are 'Just Enough Upfront documentation', 'Executable Knowledge', and 'using Automated Text Analytics' to help record, substantiate, manage and retrieve design decisions in the aforementioned phases. The main characteristic of 'Just Enough Upfront Documentation' is that knowledge required upfront includes shaping thoughts/ideas, a codified interface description, and a plan. For building the software and making maximum use of progressive insights, updating the specifications is sufficient. 'Executable Knowledge' refers to any executable artifact except the source code. Primary artifacts include Test Driven Development methods and infrastructure-as-code. A third novel approach concerns 'Automated Text Analysis' using Text Mining and Deep Learning [5], [6] to retrieve design decisions automatically.

- Nazaruka is focusing on security requirements specification and tracing methods by using a topological functioning model [7]. This research aim is to determine the way how security requirements can be specified and traced with the topological functioning modeling. The topological functioning model is a formal mathematical model that can be used as a reference model for functional and non-functional requirements of the system, which means that it can also serve as a reference model for security requirements. The author mentioned that the specification and traceability of security requirements invariably remain a challenge since modeling and analysis of security aspects of systems require additional efforts at the very beginning of the software development process.
- Ņikitenko, Bāliņa, Dubrovskis, Andersone, and Birzniece are developing the application of an IT support model for precision livestock farming in the poultry industry. The article presents a practical approach to bird weight data processing and augmentation to enable production outcome forecast model training, which depends on data sample size and quality. The authors suggest using a parametrized model, where parameter values are found through genetic optimization and thus are closely corresponding to the measured values. The proposed approach is implemented as a stand-alone software system, exposing the models through containerized services [8] enabling different use scenarios.

References

[1] L. Bass, P. Clements, and R. Kazman, *Software Architecture in Practice*. Addison Wesley, 2013.

[2] A. Zimmermann, R. Schmidt, and L. C. Jain, *Architecting the Digital Transformation*. Springer, 2021. Available: https://doi.org/10.1007/978-3-030-49640-1

[3] B. Marcinkowski, and B. Gawin, "Data-Driven Business Model Development – Insights from the Facility Management Industry," *Journal of Facilities Management*, vol. 19, no. 2, pp. 129–149, 2021. Available: https://doi.org/10.1108/JFM-08-2020-0051

[4] T. Theunissen, U. van Heesch, and P. Avgeriou, "A mapping study on documentation in Continuous Software Development," *Information and Software Technology*, vol. 142, p. 106733, 2022. https://doi.org/10.1016/j.infsof.2021.106733

[5] I. Goodfellow, Y. Bengio, and A. Courcille, *Deep Learning*. MIT Press, 2016.

[6] S. Russel and P. Norvig, *Artificial Intelligence: A modern approach*. Pearson, 2002.

[7] J. Osis and E. Asnina, "Topological Modeling for Model-Driven Domain Analysis and Software Development: Functions and Architectures," in *Model-Driven Domain Analysis and Software Development: Architectures and Functions*, Hershey, IGI Global, pp. 15–39, 2011. Available: https://doi.org/10.4018/978-1-61692-874-2.ch002

[8] K. Cochrane, J. S. Chelladhurai, and N. K. Khane, *Docker Coobook*. Packt 2018.