Formal Methods for a Digital Industry
Industrial Day at ISoLA 2021

Falk Howar[1](✉), Hardi Hungar[2], and Andreas Rausch[3]

[1] Dortmund University of Technology and Fraunhofer ISST, Dortmund, Germany
falk.howar@tu-dortmund.de

[2] German Aerospace Center, Braunschweig, Germany
hardi.hungar@dlr.de

[3] Clausthal University of Technology, Clausthal-Zellerfeld, Germany
andreas.rausch@tu-clausthal.de

Abstract. The industrial track at ISoLA 2021 provided a platform for presenting industrial perspectives on digitalization and for discussing trends and challenges in the ongoing digital transformation from the perspective of where and how formal methods can contribute to addressing the related technical and societal challenges. The track continued two special tracks at ISoLA conferences focused on the application of learning techniques in software engineering and software products [4], and industrial applications of formal methods in the context of Industry 4.0 [3,7]. Topics of interest included but were not limited to Industry 4.0, industrial applications of formal methods and testing, as well as applications of machine learning in industrial contexts.

1 Introduction

In 2011, The Wall Street Journal published Marc Andreessen's essay "Why Software Is Eating the World" in which the author predicted the imminent digital transformation of the world's economies and societies [2]. In the nearly ten years that have passed since publication, the scope and impact of that transformation has only become bigger. Without any doubt, the infrastructure of the twenty-first century is defined by software: software is the basis for almost every aspect of our daily live and work: communication, banking, trade, production, transportation - to name only a few. This has led to a situation in which for many industrial and manufacturing companies with no particular background in software, software crept into processes and products - first slowly then with an ever increasing pace and scope, culminating in the mantra that "every company needs to become a software company". From a technological perspective, the current situation is defined by a number of transformative technological innovations:

Ubiquitous Compute and Connectivity. IoT Devices and 5G technology will make it possible to put computing power and sensory equipment everywhere and connect decentralized computing power in huge distributed, heterogeneous architectures, spanning IoT devices, edge-resources, and cloud platforms. The main novel capabilities of such cyber-physical systems are (a) the

T. Margaria and B. Steffen (Eds.): ISoLA 2021, LNCS 13036, pp. 71–76, 2021.
https://doi.org/10.1007/978-3-030-89159-6_5

distributed observation, analysis, and processing of data, and (b) decentralized control of the physical world.

Data Becomes a Primary Resource. The systems described above will be used as a basis for establishing digital twins of physical devices (machines in production, harvesting equipment, supply-chains, etc.). Digital twins will act as proxies (observing and controlling physical machines) and enable the digitalization and automation of most processes and new services and features that are based on data, e.g., predictive maintenance, an autonomous harvest, or a fully synchronized supply-chain.

Machine Learning as the Basis of Applications. Applications like the aforementioned ones require the analysis of data, the discovery of patterns, and autonomous reactions to observed situations. Such features are typically realized with the help of machine learning technology, i.e., data is used to train a system instead of programming the system. The scope and complexity of learned applications is projected to increase dramatically over the next couple of years (cf., e.g., German AI strategy [13,14]).

Virtualization. Digital twins will not only act as proxies for physical devices (think shadows in the used analogy) but will become valuable assets in their own right. Processes and methods can be developed in virtual reality based on virtual twins (more precisely based on models obtained through the systems that enable digital twins): Calibration of processes, configuration of assembly lines, optimization of harvesting strategy can be computed in simulations, minimizing resources and ramp-up times, and maximizing yield in the physical world.

These trends will have an impact on virtually every enterprise. Potential cost-reductions and new services and products are expected to disrupt entire industries. This expectation has produced mantras like “Uber yourself before you get Kodaked”, alluding to Uber’s transformation of the taxi business and to Kodak’s going out of business after not pursuing digital photography as one of the early companies in that market. The challenge faced by companies is to not miss key technological opportunities while being forced to take decisions and make investments without a full understanding of the exact impact. Additionally, the discussed technological innovations are associated with new challenges and specific risks, some examples of which are:

Safety and Security. Big heterogeneous, distributed, and networked systems have many attack vectors: A plethora of libraries, frameworks, and basic systems lead to a vast space of possible configurations and combinations of software stacks on individual devices. Moreover, being connected to the Internet makes these systems easy to attack. Systems that control machines in the vicinity of humans are safety-critical—the new quality of safety-related risks in these systems originates in their openness and in the new relation of safety and security.

(Data) Eco-Systems. The full potential for value-creation of data-centric applications oftentimes cannot be realized in classical value-chains but requires eco-systems [9]. The most obvious example of such a new business model may be so-called app stores that open platforms of a vendor (classically mobile phones) to app vendors, adding value for customers through third party apps. These new business models require a degree of openness and collaboration that is not easily organized between companies that are otherwise competitors. Moreover, it is often unclear a priori which business models will be profitable in the end. For the "pure" software companies and VC culture of Silicon Valley it is easy to simply try and adapt. For a manufacturing company with long-lived physical products and processes such an agility can probably not be achieved as easily.

Machine Learning as an Engineering Discipline. In traditional software development, a set of practices known as DevOps have made it possible to ship software to production in minutes and to keep it running reliably. But there is a fundamental difference between machine learning and traditional software development: Machine learning is not just coding, it is coding plus data. Data engineering does provide important tools and concepts that are indispensable for succeeding in applying machine learning in production. Practices from DevOps and data engineering need to be integrated into an engineering discipline for ML-based software.

Quality Assurance for Machine Learning. Quality assurance of distributed applications that rely on Machine Learning as a design principle is an open challenge—scientifically and engineering-wise. A classical safety argumentation (or case) starts with a high-quality requirements specification, which should ideally be correct and complete. This specification is later used as main input for testing and verification. For the development of an AI-based system, a huge data collection is used to partially replace a formal requirements specification. This data collection is incomplete, biased, and may even contain a small percentage of incorrect data samples. In a sense, AI-based systems are "machine-programmed" using training data selected by engineers. Safety assurance then has to be based on guarantees on the quality of training data and on rigorous testing of relevant application scenarios. Such methods are being researched today but are still far from being standardized or available in certification processes.

These challenges have a tremendous impact on the engineering of software systems. The security-related essential requirement of frequent system updates, e.g., does affect architectural decisions and development processes, requiring iterative improvements during the whole software life cycle, including during operation. At the same time, the amount of data that can be obtained during operation at a massive scale by far exceeds what can be processed or stored cost-efficiently, making a purely agile development approach or blunt re-training of learned models (which are frequently hailed as silver bullets today) infeasible. System architectures as well as business models must be carefully planned and evaluated before making major investments.

Complexity and uncertainty lead to a situation in which companies wait for others to make the first move or start following hyped trends and buzz words (agile, data-lakes, or social intranet, to name only a few) instead of making informed decisions. What is required, is a software engineering discipline that allows companies to move deliberately towards their digital transformation in the face of uncertainty about future eco-systems, business models, software- and system-architectures, and applications. The aforementioned mantra to “Uber yourself before you get Kodaked” is not to be taken literally in this respect: it ironically uses Uber, a company without a sustainable business model that is alleged to exploit employees and is banned in many European countries, as a symbol of a successful transformation of an industry. Instead, the case of Uber can rather be seen as an indication of the need for a holistic approach to software engineering and digital transformation: an approach that does not simply aim at disruption but also includes a societal perspective, aims for sustainable business models, and supports sound financial and technical planning.

Formal methods can be one crucial enabler for building and scaling the software infrastructure sketched above: constructive techniques will enable systems that are correct by construction, formal verification can deliver guarantees on existing systems. Domain-specific languages may enable the seamless application of formal methods at multiple levels of abstraction. Eventually, formalization can hopefully enable (automated) alignment and integration of systems and automation of processes and quality control. Formal methods are, however, far from being able to do that today and will have to be made into enablers through the development of tools, by finding beneficial applications, and by leveraging domain knowledge. The industrial track aims at bringing together practitioners and researchers to name challenges, to look for potential contributions, to outline approaches, to name useful tools and methods, and to sketch solutions.

2 Contributions

The track featured seven contributions with accompanying papers. Contributions focused on software-enabled knowledge management and business engineering, use cases of simulation in the context of developing autonomous systems, modeling languages for industrial automation, and machine learning techniques for the assisting the assessment of data quality.

2.1 Software-Enabled Business Engineering

The paper *“Agile Business Engineering: From Transformation Towards Continuous Innovation”* by Barbara Steffen, Falk Howar, Tim Tegeler, and Bernhard Steffen [11] presents the results of a qualitative study of analogies and differences in business engineering and software engineering, starting from the observation that a high degree of automation through purpose-specific tools and other “DevOps” techniques usually has a positive impact on outcomes in innovative software projects. The authors investigate whether a meaningful analogy can be drawn to challenges in business engineering during a digital transformation.

The paper *"Towards Living Canvases"* by Barbara Steffen, Stephen Ryan, Frederik Möller, Alex Rotgang, and Tiziana Margaria [12] (in this volume) presents a proposal for improving the quality of information that is gathered in the early stages of projects (e.g., elicitation of requirements from stakeholders) by providing tools that semantically integrate information pertaining to different aspects of a system or business. The authors report on a small initial study in which multiple canvases (usually used in pen-and-paper mode) were implemented and successfully integrated.

2.2 Simulation-Based Testing of Software for Autonomous Systems

The paper *"Use Cases for Simulation in the Development of Automated Driving Systems"* by Hardi Hungar [5] (in this volume) explores potential benefits, limitations, and open challenges in the application of simulation (i.e., an inherently incomplete and inaccurate technique) during the development and validation of autonomous systems.

The contribution *"Simulation-based Elicitation of Accuracy Requirements for the Environmental Perception of Autonomous Vehicles"* by Robin Philipp, Hedan Qian, Lukas Hartjen, Fabian Schuldt, and Falk Howar [10] (in this volume) presents one concrete use case for simulation in the development of autonomous vehicles: the elicitation of formal accuracy requirements for the integration of different components (perception and planning) of an autonomous driving function.

2.3 Domain-Specific Languages for the Industry 4.0

The paper *"DSLs and middleware platforms in a model driven development approach for secure predictive maintenance systems in smart factories"* by Jobish John, Amrita Ghosal, Tiziana Margaria, and Dirk Pesch [6] (in this volume) presents a result from a case study in industrial automation in which a language workbench has been used to design a tailored domain-specific language for modeling secure predictive maintenance systems in the context of smart factories.

The contribution *"From Requirements to Executable Rules: An Ensemble of Domain-Specific languages for Programming Cyber-Physical Systems in Warehouse Logistics"* by Malte Mauritz and Moritz Roidl [8] (in this volume) presents a similar application of domain-specific languages to the one above: domain modeling and domain specific languages are used to specify and automate the behavior of a cyber-physical warehouse system that collaborates with human operators.

2.4 Applications of Machine Learning in Software Engineering

Finally, the contribution *"Mining Data Quality Rules for Data Migrations: A Case Study on Material Master Data"* by Marcel Altendeitering [1] (in this volume) reports on an industrial application of machine learning techniques for learning rules that can be used to assess the quality of data in the context of data migration.

References

1. Altendeitering, M.: Mining data quality rules for data migrations: a case study on material master data. In: Margaria, T., Steffen, B. (eds.) ISoLA 2021. LNCS, vol. 13036, pp. 178–191. Springer, Cham (2021)
2. Marc, A.: Why software is eating the world. In: The Wall Street Journal (2011/08/20) (2011)
3. Hessenämper, A., Howar, F., Rausch, A.: Digital transformation trends: Industry 4.0, automation, and AI - industrial track at ISoLA 2018. In: Leveraging Applications of Formal Methods, Verification and Validation. Industrial Practice - 8th International Symposium, ISoLA 2018, Limassol, Cyprus, November 5–9, 2018, Proceedings, Part IV, pp. 469–471 (2018)
4. Howar, F., Meinke, K., Rausch, A.: Learning systems: machine-learning in software products and learning-based analysis of software systems. In: Margaria, T., Steffen, B. (eds.) ISoLA 2016. LNCS, vol. 9953, pp. 651–654. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-47169-3_50
5. Hardi, H.: Use cases for simulation in the development of automated driving systems. In: Margaria, T., Steffen, B. (eds.) ISoLA 2021. LNCS, vol. 13036, pp. 117–128. Springer, Cham (2021)
6. Jobish, J., Amrita, G., Tiziana, M., Dirk, P.: Dsls and middleware platforms in a model driven development approach for secure predictive maintenance systems in smart factories. In: Margaria, T., Steffen, B. (eds.) ISoLA 2021. LNCS, vol. 13036, pp. 146–161. Springer, Cham (2021)
7. Margaria, T., Steffen, B. (eds.): Leveraging Applications of Formal Methods, Verification and Validation: Discussion, Dissemination, Applications, ISoLA 2016. LNCS, vol. 9953. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-47169-3
8. Malte, M., Moritz, R.: From requirements to executable rules: an ensemble of domain-specific languages for programming cyber-physical systems in warehouse logistics. In: Margaria, T., Steffen, B. (eds.) ISoLA 2021. LNCS, vol. 13036, pp. 162–177. Springer, Cham (2021)
9. Boris, O., et al.: Data Ecosystems: Conceptual Foundations, Constituents and Recommendations for Action (2019). http://publica.fraunhofer.de/documents/N-572093.html. Accessed September 2021
10. Robin, P., Hedan, Q., Lukas, H., Fabian, S., Falk, H.: Simulation-based elicitation of accuracy requirements for the environmental perception of autonomous vehicles. In: Margaria, T., Steffen, B. (eds.) ISoLA 2021. LNCS, vol. 13036, pp. 129–145. Springer, Cham (2021)
11. Barbara, S., Falk, H., Tim, T., Bernhard, S.: Agile business engineering: from transformation towards continuous innovation. In: Margaria, T., Steffen, B. (eds.) ISoLA 2021. LNCS, vol. 13036, pp. 77–94. Springer, Cham (2021)
12. Barbara, S., Stephen, R., Frederik, M., Alex, R., Tiziana, M.: Towards living canvases. In: Margaria, T., Steffen, B. (eds.) ISoLA 2021. LNCS, vol. 13036, pp. 95–116. Springer, Cham (2021)
13. The German Federal Government. Strategie Künstliche Intelligenz der Bundesregierung (2018). https://www.bmwi.de/Redaktion/DE/Publikationen/Technologie/strategie-kuenstliche-intelligenz-der-bundesregierung.html. Accessed September 2021
14. The German Federal Government. Strategie Künstliche Intelligenz der Bundesregierung - Fortschreibung (2020). https://www.bmwi.de/Redaktion/DE/Publikationen/Technologie/strategie-kuenstliche-intelligenz-fortschreibung-2020.html. Accessed September 2021