Conceptualizing Digital Twins

Romina Eramo, Francis Bordeleau, Benoit Combemale, Mark van den Brand, Manuel Wimmer, Andreas Wortmann

HAL Id: hal-03466396
https://hal.inria.fr/hal-03466396

Submitted on 5 Dec 2021

Conceptualizing Digital Twins

Romina Eramo*, Francis Bordeleau†, Benoit Combemale‡,
Mark van den Brand§, Manuel Wimmer¶, Andreas Wortmann‖

*Univeristy of L'Aquila, Italy, romina.eramo@univaq.it †École de techonologie supérieure (ETS), Canada, francis.bordeleau@etsmtl.ca
‡University of Rennes 1, France, benoit.combemale@irisa.fr §Eindhoven University of Technology, The Netherlands,
m.g.j.v.d.brand@tue.nl ¶Johannes Kepler University Linz, Austria, manuel.wimmer@jku.at ‖University of Stuttgart, Germany,
andreas.wortmann@isw.uni-stuttgart.de

Abstract—Digital Twins are an emerging concept which is gaining importance in several fields. It refers to a comprehensive software representation of an actual system, which includes structures, properties, conditions, behaviours, history and possible futures of that system through models and data to be continuously synchronized.

Digital Twins can be built for different purposes, such as for the design, development, analysis, simulation, and operations of non-digital systems in order to understand, monitor, and/or optimize the actual system. To realize Digital Twins, data and models originated from diverse engineering disciplines have to be integrated, synchronized, and managed to leverage the benefits provided by software (digital) technologies. However, properly arranging the different models, data sources, and their relations to engineer Digital Twins is challenging. We, therefore, propose a conceptual modeling framework for Digital Twins that captures the combined usage of heterogeneous models and their respective evolving data for the twin's entire life-cycle.

Index Terms—Digital Twin, Conceptual Modeling Framework, Model Types, Data and Models

I. Introduction

Digital Twins (DTs) are appearing everywhere, from agriculture, construction, engineering, production, medicine, to mention just a few prominent examples. Research and industry have produced various definitions [9], [11], ranging from underspecified (digital replica or virtual counterpart), over narrow (virtual representation based on augmented reality technology), to utopian (complete digital representation) approaches. The concepts found in literature range between (a) high-fidelity design-time models used for design-space exploration, dimensioning, or validation, and (b) software systems used to monitor, comprehend, and optimize the behavior of another system during its runtime.

A DT is a virtual representation (or replica) of an Actual System (AS) that is continuously updated with real-time data throughout its life-cycle and, at the same time, can interact with and influence the AS [2]. DTs can be built for a variety of purposes, such as for the design, development, analysis, simulation, and operations of non-digital systems in order to understand, monitor, and/or optimize the AS. Many domains employ DTs to better understand, control, and optimize the behavior of complex systems, either during their design-time (e.g., for design-space exploration) or at runtime (e.g., to improve performance/productivity or prevent failures). Hence, DTs are becoming an important software engineering tool to harness the complexity of software engineering in a wide range of application domains. To this effect, we consider a DT of an AS to comprise a set of models of that AS and provides a set of services that use data, possibly obtained from the AS, and models for specific purposes with respect to the AS [1].

The complexity of systems grows fast and models have become crucial to understand them. Consequently, today's sophisticated systems are engineered with models from different engineering domains. Hence, also their DTs need to integrate various heterogeneous models to address the different aspects of the system. These models may be engineering models (e.g., BIM, Modelica, Simulink, mathematical equations) or software models (e.g., AADL, MontiArc, SysML, UML, or models in some kind of DSL).

To make sense of these models during a system's design-time and its runtime, they must be connected to data obtained from the AS and its environment. The models and data enable implementing services related to the AS. How the different models, data, and services of a DT relate depends on its purpose and application.

We investigated different characterizations of DTs [9], [11] to distill their essentials in a conceptual modeling framework focused on making data, models, and relations explicit. To this end, we present the framework and its instantiations aiming to support conceptualizing DTs and systematically engineering novel DT applications. The framework details the core parts of DTs and its instantiations are blueprints for software engineers that describe which DT parts they need to provide, interconnect, and integrate to achieve different DT utilizations.

II. A Conceptual Framework for Digital Twins

We consider a DT to be comprised of a set of models and data associated with the AS that enables the creation of a set of *services* [1], corresponding to the functionalities provided by the DT. For instance, the DT can use simulation (even in communication with other DTs) helping decision-making for maintenance or improvement of the AS based on new requirements and/or available resources.

Fig. 1 presents a conceptual framework for DTs described using the MODA framework [3], a conceptual modeling [13] framework that aims at supporting the description of data-centric systems in terms of models, data, and transformations. By proposing a conceptual framework, we do not discuss particular tools or technologies for realizing DTs, but we categorize the different roles and the relationships of artefacts on a conceptual level. By this, we follow but also extend the viewpoints of existing frameworks proposed in the literature [9], [11].
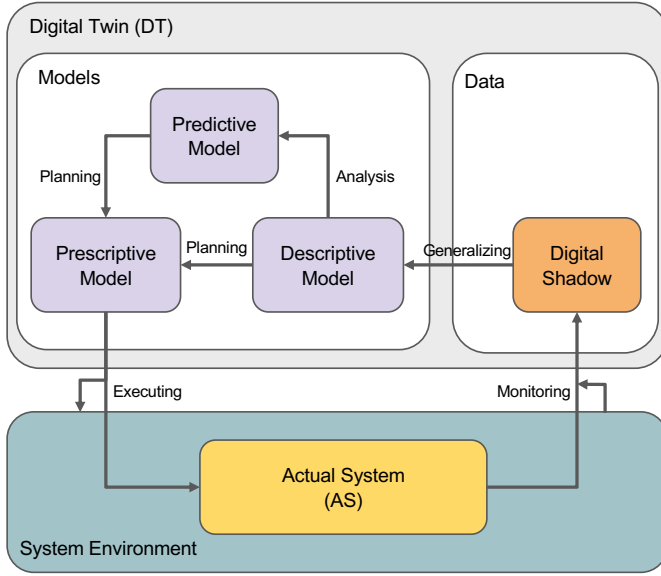
Fig. 1: Conceptual framework for DTs based on MODA [3]

A DT is a virtual representation of an actual system (AS) that exists within its environment. The AS produces data, which is related to different aspects of the system, and the DT captures this data and uses a set of models to conduct different types of operations/actions on the AS (as we will see later, the selection of the models depends on the purpose of the DT). The main components of the framework we are proposing are as follows.

- **Actual system and its environment**. The AS refers to the system associated with the DT. Collecting and storing as well as calculating and inferring data—depending on the system itself—is mandatory for a DT, that should capture the relevant aspects, including the required features and relationships, of the AS with respect to the contexts and environments (System Environment) in which it operates [11]. The AS is an essential element for the existence of the DT itself.
- **Data**. This component is related to the storage and representation of the current and past data of the AS relevant for the DT. Data and information are needed to represent the AS in the specific digital models of the DT. A DT comprises different types of data flows (see the arrow labeled Monitoring in Fig. 1): data obtained from monitoring and sensing, measured data, and external/historical data from the system or its environment (e.g., environmental data, technical data, constraints, etc.). To this end, DTs comprise Digital Shadows, i.e., purposefully abstracted and aggregated data structures [1], that represent one-way data flow between the state of the AS and its digital representation.
- **Models**. This component is represented by one or more models of the system, or its parts [9]. Models address different aspects and disciplines of a system (e.g., engineering models, software models, scientific models) and can be of various language types [2]. As defined in the MODA framework [3], we identify three roles that models can play in a DT: Descriptive Model,

Predictive Model, and Prescriptive Model. A Descriptive Model reflects the system or the system's environment in a descriptive manner, representing current or past aspects of the actual system, facilitating understanding, and enabling analysis [3]. The arrow labeled Generalizing (between Digital Shadow and Descriptive Model) represents the application of techniques that generalize the different kinds of data to yield (or calibrate) a descriptive model. In addition, models may also reflect the actual or future system behaviour in a predictive or prescriptive manner. A Predictive Model is used to predict information that has not been measured, allowing decision-making and trade-off analyses. This can include models for analysis (e.g., Petri nets), simulation (e.g., Simulink), and machine learning (e.g., different types of neural networks). In contrast, a Prescriptive Model is a description of the system to be realized, driving the constructive process, including runtime evolution in the case of self-adaptive systems [3]. The arrows labeled Analysis and Planning represent decision support activities (e.g., what-if analysis) in order to feed and update the prescriptive model. Finally, and to close the loop between DT and AS, the arrow labeled Executing involves deploying and executing the actions on the AS and/or its environment, based on the prescriptive model.

III. DIGITAL TWIN APPLICATIONS

Research and practice have produced various DT implementations. We identified a representative set of DT applications that involve different stages of a system life-cycle, notably design, manufacturing, maintenance and utilization [14]. In the following, we characterize these applications using our conceptual framework. Fig. 2 illustrates the set of DT applications as instances of the DT conceptual framework.

A. Design

The design of new systems requires the development team to collaborate with stakeholders through exchanging and sharing design data and relevant information [14] to investigate alternatives and validate design decisions. In application domains like aerospace, automotive, construction, health care, and manufacturing, DTs are used during the system design phase to improve/optimize the system by enabling different types of analysis and simulation that allows exploring different design alternatives based on actual data.

In such cases, data is collected both from the System being designed and from the Environment (Stakeholders). It can relate system functions and appearance, technical and process data, tests results from the design process, and stakeholders/customer feedback. This data is recorded and managed by the DT using a set of Digital Shadows, each focused on data related to different design aspects.

On the models side, this data, together with historical data of similar systems, is used to produce different types of Descriptive Models, e.g., feature models, structure models, behavior models, and domain models. Beside allowing to better understand different aspects of the system design, these
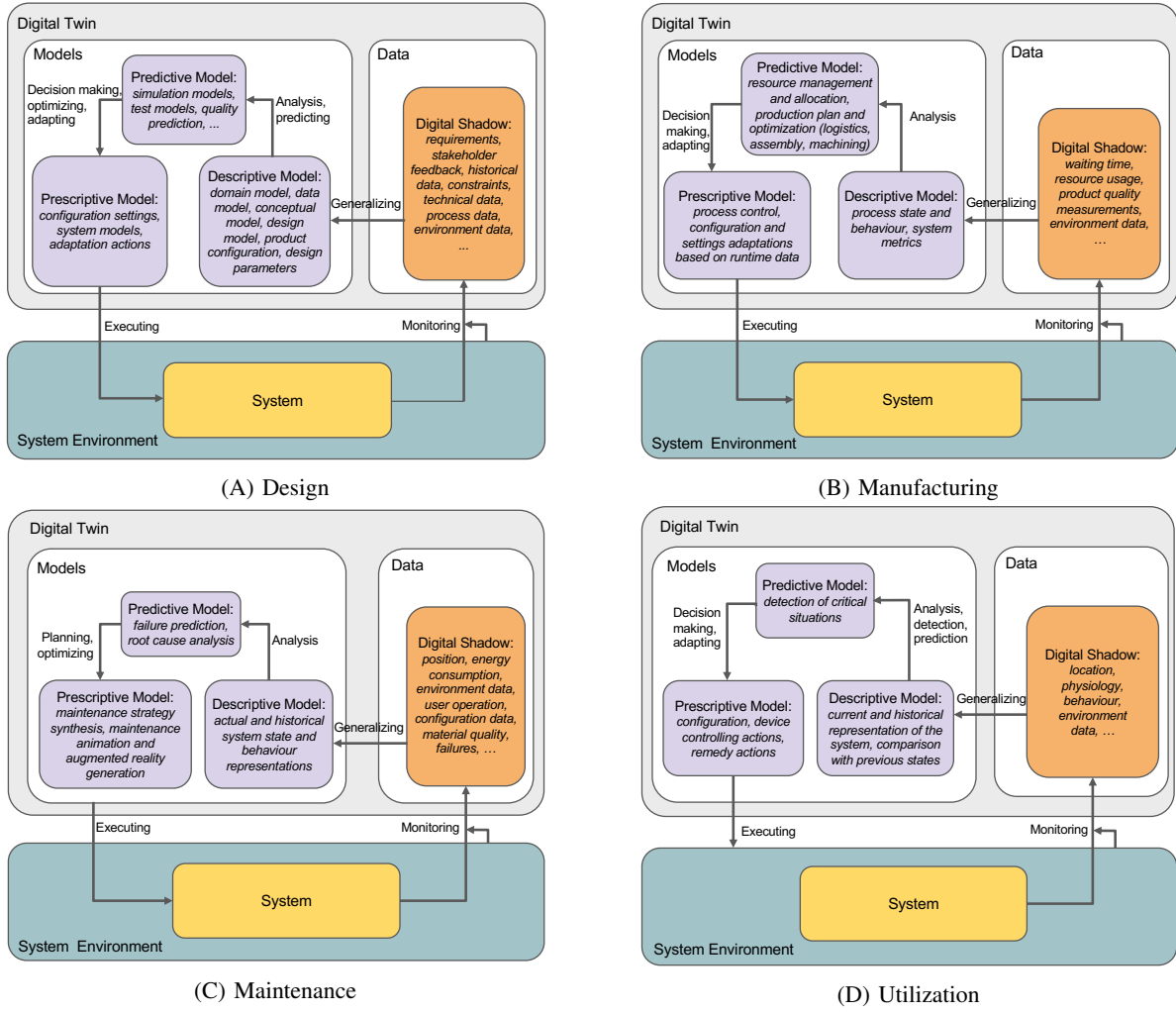
Fig. 2: Instances of the DT conceptual framework representing different DT applications

models are used to produce Predictive Models that enable different types of analysis, simulation, and tests. For example, a predictive model can be used to simulate different scenarios to allow to accurately predict specific properties related to performance. Finally, decisions to improve the system design (e.g., configuration, adaptation, and refactoring) are encoded in Prescriptive Models and then applied in the system design.

B. Manufacturing

Manufacturing refers to the overall process used to transform raw materials into finished products. DTs are used in domains such as Industry 4.0, processor manufacturing, and automotive, to help optimizing different aspects of the manufacturing process, including production flow, product quality, resource management, waste reduction, and maintenance intervals [15].

In this context, the DT monitors different types of data related to both the Manufacturing System and its environment. It typically includes time required at the different production phases, waiting time, resource usage, different measures related to product quality, temperature, and humidity.

On the modeling side, Descriptive Models are used to extract different types of process information and metrics from

the data, e.g., lead time, percentage of productive time, and different quality metrics. Descriptive Models are also used to accurately describe the manufacturing system (structure, behavior, and state), using different types of system modeling techniques, and the process in the form of process models [12] and Value Stream Maps (VSM) [6]. Then, Predictive Models can be produced to analyze and simulate the different aspects of the manufacturing process to improve/optimize it. Once decisions are taken based on the results of the predictive models, Prescriptive Models are produced and used to drive the execution of the decided strategies to improve the product manufacturing system.

C. Maintenance

Complex systems (e.g., aircraft, robots, automobile, and electric power equipment) comprise complex structures, heterogeneous components and materials, and inconsistent degradation and malfunctioning, impacting, for instance, safety and performance. Whereas, static, probabilistic, and heuristic-based methods are no longer sufficient, with the DT methodology, relevant services for the analysis of the system and the prediction of degradation and anomalous events can be

provided, which allows to move to the area of predictive maintenance [14].

Real-time state data is transmitted to the Digital Shadow to realize the synchronous linkage between the actual system and the corresponding DT as well as to store historical data. Data may include position, energy consumption, environment data, user operation, configuration data, execution information, material information, failures, etc.

The Descriptive Model is built on top of the real-time data and the obtained historical data. Through running relevant failure prediction algorithms, e.g., based on machine learning models, system failure prediction (Predictive Model) is possible. The output of the failure prediction is passed to a Prescriptive Model able to synthesize the maintenance strategy to be deployed on the physical product, e.g., by applying AI planning algorithms to consider also constraints from the product environment such as currently processed orders etc.

D. Utilization

Not only maintenance, but also the utilization of systems in a certain context is currently a topic for DTs. For instance, DT technologies emerged in domains such as agriculture/farming [7] to improve/optimize natural, non-engineered, systems. A main characteristic of these systems is that they cannot be directly controlled. The DT represents the system and its environment, monitors and indirectly controls and interacts with the system by modifying its environment which is equipped with automation components. An example of such a system is a greenhouse used to grow vegetables. The vegetables themselves, of course, cannot be controlled directly, but a DT can control automation components for influencing light, temperature, and humidity to optimize the growth of the vegetables [5]. Another example is a medical treatment system that can monitor different health aspects of a patient to improve/optimize treatments for different health conditions [4], [10].

Concerning the later application domain, several kinds of data are monitored and represented by the Digital Shadow, e.g., physiology data retrieved from body sensors, location data retrieved from motion sensors, and behavioral data extracted by image analysis from cameras [11].

Such digital shadows are then used to build the Descriptive Model, that can be analyzed and employed to identify certain risks and benefits the system is experiencing. For instance, the output of the analysis can be a Predictive Model for the detection of critical situations to be directly warned to the system. To counteract such critical situations may require reconfiguration actions or further remedy actions, derived from a Prescriptive Model, which can be automatically deployed on the automation components to positively influence the system. Please note that this application can be virtually replicated, for instance, to explore if a specific situation may occur and how it may be prevented by certain counter-measures.

IV. Lessons Learned and Looking Ahead

The identification and description of the presented architectures of DT applications based on the conceptual modeling framework characterizes the current state-of-the-art in DTs, especially related to Industry 4.0. It provides a language to speak and discuss about DTs, especially, from a software engineering perspective. In addition, it allows the re-use of concepts across DTs, because many DTs will be based on hybrid architectures, i.e., a mix of the presented variants. The list of DT applications presented here is not exhaustive, but shows the usefulness of the presented framework and the described DT applications show the appropriateness of its foundation, the MODA framework.

Mature DTs will have certainly elements of self-adaptiveness in combination with machine learning, simulation, and data processing elements. Thus, DTs may be classified as MAPE-K [8] systems which require also new architectural styles to combine models and data as well as different computation mechanisms such as simulation, machine learning, planning reasoning, and data analytics. Moreover, When using the proposed framework in concrete cases with specific modeling techniques, and associated tools, specific integration issues need to be addressed, as discussed in [2] which opens up an important research line.

More (literature) research needs to be done to identify other architectural patterns or combinations of these patterns which may be documented in a pattern catalogue for DTs. Also, explicitly defining the interface to communicate with the data and the different models would support the development of generic services to be reused across DTs such as it is already provided for simulators realizing the Functional Mockup Interface (FMI) standard. Finally, the conceptual elements have to be mapped to concrete technologies which would allow to deploy DTs on dedicated platforms with higher automation potential.

References

[1] P. Bibow, M. Dalibor, C. Hopmann, B. Mainz, B. Rumpe, D. Schmalzing, M. Schmitz, and A. Wortmann. Model-driven development of a digital twin for injection molding. In Proc. of the 32nd International Conference on Advanced Information Systems Engineering (CAiSE), volume 12127 of Lecture Notes in Computer Science, pages 85–100, 2020.

[2] F. Bordeleau, B. Combemale, R. Eramo, M. van den Brand, and M. Wimmer. Towards model-driven digital twin engineering: Current opportunities and future challenges. In Proc. of First International Conference on Systems Modelling and Management (ICSMM), volume 1262, pages 43–54, 2020.

[3] B. Combemale, J. Kienzle, G. Mussbacher, H. Ali, D. Amyot, M. Bagherzadeh, E. Batot, N. Bencomo, B. Benni, J. Bruel, J. Cabot, B. C. Cheng, P. Collet, G. Engels, R. Heinrich, J. Jezequel, A. Koziolek, S. Mosser, R. Reussner, H. Sahraoui, R. Saini, J. Sallou, S. Stinckwich, E. Syriani, and M. Wimmer. A hitchhiker's guide to model-driven engineering for data-centric systems. IEEE Software, 38(4):71–84, 2021.

[4] H. Elayan, M. Aloqaily, and M. Guizani. Digital Twin for Intelligent Context-Aware IoT Healthcare Systems. IEEE Internet of Things Journal, pages 1–9, 2021.

[5] A. Ghandar, A. Ahmed, S. Zulfiqar, Z. Hua, M. Hanai, and G. Theodoropoulos. A decision support system for urban agriculture using digital twin: A case study with aquaponics. IEEE Access, 9:35691–35708, 2021.

[6] P. Hines and N. Rich. The seven value stream mapping tools. International Journal of Operations & Production Management, 17(1):46–64, 1997.

[7] S.-K. Jo, D.-H. Park, H. Park, and S.-H. Kim. Smart livestock farms using digital twin: Feasibility study. In Proc. of the International Conference on Information and Communication Technology Convergence (ICTC), pages 1461–1463. IEEE, 2018.

[8] J. Kephart and D. Chess. The vision of autonomic computing. Computer, 36(1):41–50, 2003.

[9] W. Kritzinger, M. Karner, G. Traar, J. Henjes, and W. Sihn. Digital twin in manufacturing: A categorical literature review and classification. IFAC-PapersOnLine, 51(11):1016–1022, 2018. 16th IFAC Symposium on Information Control Problems in Manufacturing INCOM 2018.

[10] Y. Liu, L. Zhang, Y. Yang, L. Zhou, L. Ren, F. Wang, R. Liu, Z. Pang, and M. J. Deen. A novel cloud-based framework for the elderly healthcare services using digital twin. IEEE Access, 7:49088–49101, 2019.

[11] R. Minerva, G. M. Lee, and N. Crespi. Digital Twin in the IoT Context: A Survey on Technical Features, Scenarios, and Architectural Models. Proceedings of the IEEE, 108(10):1785–1824, 2020.

[12] J. Recker, M. Rosemann, M. Indulska, and P. Green. Business process modeling-a comparative analysis. Journal of the Association for Information Systems, 10(4):333–363, 2009.

[13] D. Schmidt. Guest Editor's Introduction: Model-Driven Engineering. Computer, 39(2):25–31, 2006.

[14] F. Tao, J. Cheng, Q. Qi, M. Zhang, H. Zhang, and F. Sui. Digital twin-driven product design, manufacturing and service with big data. International Journal of Advanced Manufacturing Technology, 94:3563–3576, 2018.

[15] F. Tao, Q. Qi, L. Wang, and A. Nee. Digital Twins and Cyber–Physical Systems toward Smart Manufacturing and Industry 4.0: Correlation and Comparison. Engineering, 5(4):653–661, 2019.
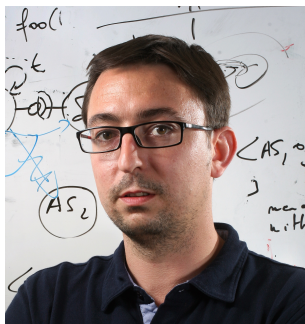
VITAE

Romina Eramo is assistant professor at the University of L'Aquila (Italy). Her research interests include model-driven engineering, software quality, continuous software engineering, DevOps, and digital twins. She is involved in several program committees of international conferences, reviewing activities and conference organization. She published several articles in journals and proceedings of international events on her topics. She has been working and leading different European and Italian research projects. For more information, please visit https://people.disim.univaq.it/romina.eramo/.

Francis Bordeleau is professor at École de technologie supérieure (ETS) in Montréal. He currently holds the ETS Kaloom-Telus Industry Research Chair in DevOps. He has more than 25 years of experience in software engineering in industrial and academic environments. His main research interests include model-based engineering, software processes, DevOps, digital twins, and open source software. Prior to joining ETS in 2018, he was the founder of Cmind (2017-2020), product manager of SW Development at Ericsson (2013-2017), founder and CEO of Zeligsoft (2003-2013), director of Tooling Business for PrismTech (2010-2013), and assistant professor at Carleton University (1997-2006). He has also been member of the organizing and program committee of many international conferences and workshops.For more information, please visit https://www.etsmtl.ca/en/research/professors/fbordeleau/.

Benoit Combemale is full professor of software engineering at the University of Rennes 1. He is leading the Computer Science Department at the engineering school ESIR, and involving within the research team DiverSE, joint to the IRISA and Inria labs. He is also adjunct researcher in the SM@RT team of the IRIT labs. His research interests include model driven software and systems engineering (MDE), software language engineering (SLE) and software validation & verification (V&V); mostly in the context of (smart) cyber-physical systems and Internet of things. For more information, please visit http://combemale.fr.

Mark van den Brand is a full professor of Software Engineering and Technology in the Department of Mathematics and Computer Science, and a visiting professor at Royal Holloway, University of London. His current research activities are on model driven engineering, domain specific languages, meta-modeling, model management, digital twins, and automotive software engineering. His research is industry inspired; he works with most of the high-tech companies in the Eindhoven (The Netherlands) region. He is project leader of a research project on Digital Twins.He was and is member of PCs on workshops and conferences related to software engineering, language engineering, rewriting, reverse engineering, and software maintenance. He is on the editorial board of the journals Science of Computer Programming, Open Computer Science, and Computer Languages (COLA). He is deputy Editor-in-Chief of platinum open access journal JOT.

Manuel Wimmer is full professor leading the Institute of Business Informatics - Software Engineering at the Johannes Kepler University Linz and he is the head of the Christian Doppler Laboratory CDL-MINT. His research interests comprise foundations of model engineering techniques as well as their application in domains such as tool interoperability, legacy modeling tool modernization, model versioning and evolution, and industrial engineering. For more information, please visit https://www.se.jku.at/manuel-wimmer/.

Andreas Wortmann is a professor at Stuttgart University. He is a member of the IEEE Technical Committee on Software Engineering for Robotics and Automation and a board member of the European Association for Programming Languages and Systems. He also serves on the editorial boards of the Journal of Object Technology and the Journal on Software and Systems Modeling. His research interests include model-driven engineering, systems engineering, software language engineering, and robotics. He has co-authored over 90 international conference and journal publications. For more information, visit www.wortmann.ac.