Towards Model-Driven Digital Twin Engineering: Current Opportunities and Future Challenges

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

To cite this version:

Francis Bordeleau, Benoit Combemale, Romina Eramo, Mark van den Brand, Manuel Wimmer. Towards Model-Driven Digital Twin Engineering: Current Opportunities and Future Challenges. ICSMM 2020 - International Conference on Systems Modelling and Management, Jun 2020, Bergen, Norway. hal-02946949

HAL Id: hal-02946949
https://hal.inria.fr/hal-02946949
Submitted on 23 Sep 2020

Towards Model-Driven Digital Twin Engineering: Current Opportunities and Future Challenges

Francis Bordeleau[1], Benoit Combemale[2], Romina Eramo[3],
Mark van den Brand[4], and Manuel Wimmer[5]

[1] Ecole de Technologie Superieur, Universite du Quebec, Canada
`francis.bordeleau@etsmtl.ca`
[2] University of Toulouse, CNRS IRIT `benoit.combemale@irit.fr`
[3] University of L'Aquila, Italy `romina.eramo@univaq.it`
[4] Eindhoven University of Technology, The Netherlands `m.g.j.v.d.brand@tue.nl`
[5] CDL-MINT & Johannes Kepler University Linz, Austria `manuel.wimmer@jku.at`

Abstract. Digital Twins have emerged since the beginning of this millennium to better support the management of systems based on (real-time) data collected in different parts of the operating systems. Digital Twins have been successfully used in many application domains, and thus, are considered as an important aspect of Model-Based Systems Engineering (MBSE). However, their development, maintenance, and evolution still face major challenges, in particular: (i) the management of heterogeneous models from different disciplines, (ii) the bi-directional synchronization of digital twins and the actual systems, and (iii) the support for collaborative development throughout the complete life-cycle. In the last decades, the Model-Driven Engineering (MDE) community has investigated these challenges in the context of software systems. Now the question arises, which results may be applicable for digital twin engineering as well.
In this paper, we identify various MDE techniques and technologies which may contribute to tackle the three mentioned digital twin challenges as well as outline a set of open MDE research challenges that need to be addressed in order to move towards a digital twin engineering discipline.

Keywords: Heterogeneous modeling, modeling languages, digital twins.

1 Introduction

The complexity of the new generation of systems developed in the context of IoT, Industry 4.0, digital transformation, high-tech systems, and smart systems (e.g., ranging from smart buildings over smart cities to smart mobility) triggers a set of major challenges, both from a technical and a business perspective. Organizations developing systems in these application domains must constantly be looking for ways to improve the quality and efficiency of their processes, systems, and products (in terms of different factors like costs, and energy consumption) and reduce development, operations, and maintenance cost. Moreover, these systems must be able to adapt to constantly evolving (open) contexts and environments [21].

The very essence of this new generation of systems requires maximizing the use of data collected throughout the system life cycle, which need to be processed, organized and structured to help managing and improving the systems.

Digital Twins have been emerging in various engineering disciplines since the beginning of this millennium to better manage systems based on (real-time) data collected in different parts of the systems. They have been already successfully used in many application domains and are now considered as an important aspect of Model-Based Systems Engineering (MBSE) [10]. While they were initially developed solely for physical systems (e.g., space systems at NASA [30] and industrial systems such as Industry 4.0 manufacturing components), they are now used for many other types of systems, including cyber-physical (e.g., a car or robotic system), socio-cyber-physical (e.g., a smart building, city, enterprise), or natural (e.g., a cell) systems. One of their key characteristics is that they allow leveraging the benefits provided by software (digital) technologies for the design, development, analysis, simulation, and operations of non-digital systems.

While their benefits have been demonstrated in many contexts, their development, maintenance, and evolution, trigger major challenges. In this paper, we focus on three main challenges: (i) the use and integration of a set of heterogeneous models that are required to address the different aspects and disciplines of a system, (ii) the synchronization of the digital twin with runtime data, and (iii) the co-development and management of the evolution of digital twins by teams of engineers.

In the last decades, the Model Driven Engineering (MDE) community has addressed many issues and challenges that are currently faced by other engineering disciplines in the development and evolution of complex systems based on digital twins. Among other things, the MDE community has developed solutions for the creation and evolution of modeling artifacts, support for collaborative development of large distributed teams, agile development, management of models linked to runtime systems, integration of heterogeneous models, etc. MDE is now considered a mature engineering discipline with a huge body of knowledge [13] that provides a broad range of modeling languages (e.g., UML [31], SysML [18], BPMN [3], Modelica [19], BIM [16], XES [1]), tools and methodologies addressing different development aspects.

The main objectives of this paper are to: (i) discuss how different techniques and technologies developed by the MDE community can contribute to resolve the three main challenges previously identified; and (ii) identify a set of open research challenges, related to MDE, that need to be addressed to support the development and evolution of digital twins - potentially leading to a digital twin engineering discipline.

2 Digital Twins and MDE

In this section, we describe the concept of digital twin from an MDE perspective, provide concrete examples of digital twins in different application domains (smart building/city, high tech, and smart enterprises), and discuss the three main digital twin challenges based on the given examples.

2.1 Digital Twins and their Relationship to MDE

A digital twin is defined as a virtual representation (or replica) of an actual system that it is continuously updated with real-time system data throughout its life cycle and, at the same time, allows to interact with and influence the system (cf. Fig. 1 for an illustration). A system can be associated with a set of (one or more) digital twins, each defined for a specific purpose.

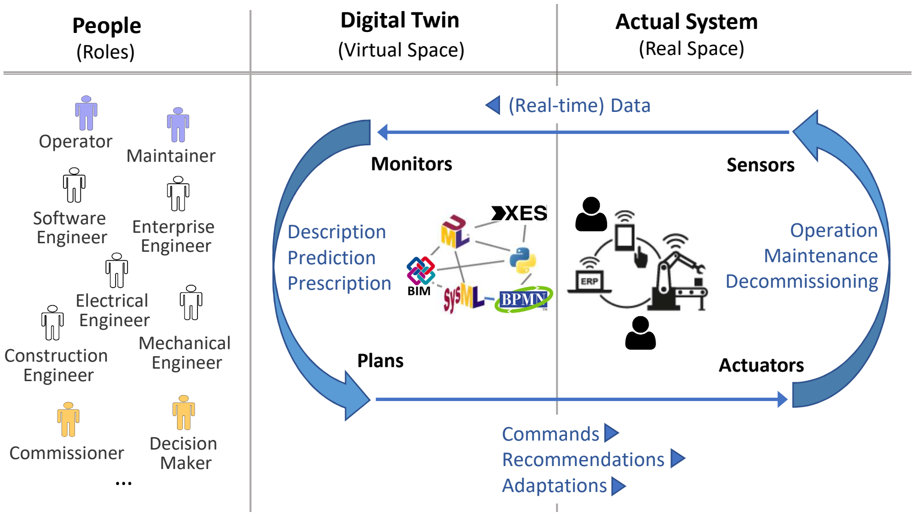

Fig. 1. Digital twin from an MDE perspective.

From a systems engineering perspective, digital twins can provide many important benefits. For example, a digital twin can be developed to enable real-time monitoring of actual systems and processes, provide timely analysis of data to intercept problems before they arise, prevent downtime, improve performance, schedule preventive maintenance to reduce/prevent faults, support decision making, plan for future upgrades and new developments, provide a mechanism to develop the control software before finalizing the physical artifact, or simulate ”bad-weather behaviour” in a safe way while the system is running.

The concept of digital twin is intimately linked to models and MDE. Each digital twin is defined in terms of one or more models of the system, or part of it [26]. The type of models used depends on the type of system and the purpose of the digital twin. For instance, models may be used for descriptive, predictive, and prescriptive purposes. Also, since the different aspects typically involve different engineering disciplines, the development of digital twins involves the definition, combination, and management of a set of heterogeneous models from various disciplines and require the involvement of teams of engineers collaborating on the development of the digital twins, and models (cf. the people swim lane of Fig. 1 for some examples).

Data play a central role in digital twins (cf. real-time data stream from the actual system to the digital twin in Fig. 1). Digital twins ”connect” models with the runtime data provided by the different sensors and measurement devices of the system with dedicated monitors. It is this connection between the models and system data that constitutes the essence of a digital twin and that enables reasoning about different aspects of a systems based on actual system data (often referred to as digital shadow). In digital twins, models are not only important for the design of the system, but also for the interpretation and analysis of the data, as well as for investigating different design or operation alternatives.

2.2 Examples of Digital Twins

Smart Buildings and Smart Cities. Digital twins can be used to manage and optimize many features of smart buildings and smart cities. As a concrete example, a digital twin was developed to help manage (reduce) the energy consumption of the Vienna Airport[6] to keep it under a certain threshold to avoid the necessity of building a new power line which would require a very high investment, and to avoid planning mistakes that would only become visible in the operation of the physical airport. The core of the Vienna Airport digital twin was built using Building Information Modeling (BIM) [16] and uses building physics simulations to reason about the energy balance of the airport.

A digital twin can use data collected from the city infrastructure to analyze and improve a smart city from several points of view [28]. For instance, it can be used to plan, inspect, and validate different aspects of a city before it is built. As an example, the new capital of the Indian state of Andhra Pradesh, was presented in [36] as the first entire city born with a digital twin. In this case, the digital twin collects data (via sensors, drones or other IoT and Industrial IoT tools) and uses a set of models to perform advanced analytics, Machine Learning (ML) and Artificial Intelligence (AI) to gain real-time insights about the physical asset's performance, operation or profitability.

High-tech Systems. In [29], a model driven approach is presented by means of a domain specific language to create digital twins in an efficient way. The concepts of wafer handler, the component in a wafer scanner that takes care of moving wafers from storage to source for exposure, are captured in a domain specific language. The goal of the digital twin of the wafer handler was to test the control software outside a clean room and to test bad weather without causing physical damage to the equipment itself. It also allowed to experiment with other set-ups and to test the performance of these set-ups.

Digital Enterprise. Enterprise modeling, such as the specification of organizational models, workflow models, business models, is used since decades and especially since the emergence of enterprise architecture. However, new advances in data processing allows for real-time monitoring of enterprises not only on a technology level but on the business level as well. Connecting runtime data to the enterprise design models allows for real-time monitoring, conformance checking, and optimizations. In particular, the analysis of runtime data by process mining techniques [34] allows to extract knowledge from historical data and to close the gap from the technological and business perspectives. Thus, organizations are starting to use so-called enterprise digital twins to coordinate the dependencies between people, business processes, and IT.[7]

2.3 Digital Twin Challenges

To be successful, digital twins need to be properly planned, resourced, and managed like any other software product. Based on the analysis of digital twins in different contexts and application domains, including the examples described in Subsection 2.2, we focused on the integration and management of heterogeneous models as a key aspect. In particular, we identified three main challenges to which MDE can contribute:

---

[6] Virtual Airport City Vienna: `https://simlab.tuwien.ac.at/virtual-airport-city-vienna/`

[7] `https://www.softwareag.com/info/innovation/enterprise_digital_twin/default.html`

1. **Systematically managing heterogeneous models.** In order to capture the different technical and domain aspects of a system, the creation of digital twins requires the development and combination of models from different engineering disciplines. As a result, a main challenge relates to the management of a collection of heterogeneous models to ensure their consistency.
2. **Bi-directional synchronization with the actual system.** The essence of digital twins is based on their ability to exploit runtime information to improve the management of a system. To achieve this, bi-directional communication and synchronization must be establish between the digital twin and the system. A main challenge consists in providing scalable solutions to support this bi-directional communication and synchronization throughout the life cycle of the system.
3. **Collaborative development throughout the system life-cycle.** The development and evolution of digital twins, and of the heterogeneous models that they are based on, require the collaboration of many engineers that can work simultaneously on different parts of the digital twins, and associated models. For this reason, it is essential that the development environment provides proper support for collaborative development.

3 MDE Contributions to Digital Twin Challenges

In this section, we discuss how MDE techniques and technologies can contribute to the three digital twins challenges described in the previous subsection.

3.1 MDE Contribution to the Management of Heterogeneous Models

As actual systems grow in complexity, also their virtual counterparts need to capture these multiple aspects of the system. Thus, they have to move from digital representations of single entities to models of interconnected components. In software engineering, there is the common trend of having multiple models describing different aspects of the software. Especially complex software systems are mostly built using several (modeling) languages. To deal with such situations, various model management tools have been developed, such as the Epsilon [23], over the last decade.

Families of languages have been proposed such as UML [31]. In such families, multiple modeling formalisms are brought together supported by multiple development methodologies, e.g., RUP (Rational Unified Process) and development environments, e.g., Papyrus, are developed to support the creation of such models.

From a software point of view, a digital twin must be able to integrate a collection of heterogeneous models, and take care of the orchestration and interaction of the models. This is very similar to the development of modern software systems. In order to ensure that the integrated models make sense from a systems engineering point of view, it is essential that the models are consistent with respect to behaviour, exchanged data, units, etc. A modeling language like SysML, the systems engineering extension of UML, supports both hardware and software development. Engineers are thus faced with the difficult task of relating information presented in different models. Although existing modeling tooling performs well on this point, establishing and ensuring the consistency between heterogeneous models is a new area to explore [14]. It involved identification of similar concepts both syntactically and semantically, and the definition of possibly complex composition operators with various possible semantics (e.g., consistency, refinement, or coordination). Addressing this challenge will support the following critical aspects of developing digital twins: communication across teams working on different aspects and engineering disciplines, coordination of work across the teams, well-defined management of the teams to ensure product quality, and a broader engagement of the final users in the loop [9]. Based on these observations, it is important to ensure that modeling concepts are semantically well understood.

3.2 MDE Contribution to the Bi-directional Synchronization between Digital Twins and Actual Systems

To enable bi-directional synchronization between digital twins and actual systems, support is required for the the following steps: (i) extracting the relevant information from the raw runtime data provided by the system, (ii) enabling the digital twin models to use the information extracted from the runtime data, and (iii) enabling the information produced by the digital twin models to be fed into the system during execution.

This set of issues has been addressed by the MDE community in the context of Models@run.time [6, 8]. Models@run.time have been developed to extend the use of software models produced during the design phase to the runtime environment with the goal of providing effective mechanisms to help reasoning about the system based on information generated by the system during its execution. Moreover, Models@run.time provide a view on a running software system that can be used for monitoring, analyzing, fixing design errors, exploring new design alternatives, or adapting it through a causal connection between a model and the system. However, having one single model@run.time that reflects a running system is not practicable when dealing with complex software systems. From a digital twin point of view, multiple and potentially diverse models@run.time are typically required to capture different system concerns [5, 35]. Furthermore, the models associated with a digital twin need to be continuously updated with runtime data throughout the system life-cycle to reflect its status (regarding different aspects like performance, energy consumption, health, and maintenance) [4].

3.3 MDE Contribution to the Collaborative Development of Digital Twins

For decades, software development is considered as a collaborative effort, and thus, has become place and time independent. This has been facilitated by introducing advanced means of communications, but also tools to store different versions of software systems and to allow developers to work simultaneously on one single software system. Over the years, the collaborative development technologies and tools initially developed for code (i.e., text-based files) have been adapted for models. This includes the use of version control technologies that somehow constitute the foundation of collaborative development, but also the use of tools to compare different versions of a model to identify differences and to support the resolution of conflicts and the merge of the different versions. Work on this topic includes [12, 24] and tools like EMFCompare [2, 33]. As a result, current MDE tools allow storing of models, performing model comparison and merge, and keeping track of changes in order to deal with co-development and model evolution. These ingredients are necessary to be able to work in a collaborative way on models. The traditional engineering disciplines have a more individual or sequential way of working and the classical tooling to support the development of models is often lacking modern support to develop models in a collaborative way as well as dedicated model evolution support. However, for realizing the vision of digital twins, these features are highly needed in order to incrementally improve systems based on digital twin updates as well as to rollback to previous versions or explore possible future versions. Finally, collaboration support is needed across engineering discipline borders which may also require dedicated collaboration models supported by tool chains in order to organize work. The availability of powerful tooling for these aspects is a prerequisite to facilitate proper systems engineering in general and digital twin development in particular.

4 Open Research Challenges

In this section, we discuss a set of open research challenges that arise from the MDE contributions in the context of digital twins presented in the previous section.

Modeling Languages for Digital Twins. Because of the specific nature of the application domains in which digital twins are used, the use of a Domain-Specific Modeling Languages (DSML) may be an option to consider. Many language workbenches have been developed and used successfully in different application domains in the last decade, e.g., EMF[8], the GEMOC Studio [11], MPS[9], Rascal [22] or Monticore [25]. However, in spite of the benefits they provide, the use of DSML requires an upfront cost (in terms of time and effort) to identify the set of domain concepts to be included in the DSML, and to define and formalize the relevant abstractions of the language. In addition, legacy languages may be already in use for decades which are not yet fully digitalized. Finding a good mix, both in terms of number of languages and formalization is a major challenge for each domain.

Architectural Framework for Digital Twins. One open research challenge relates to the development of a framework for digital twins that would allow reducing the cost of building a digital twin. This framework should be defined in terms of a basic architecture, and set of language concepts and services, for instance for integrating existing heterogeneous models, that can be adapted and extended for specific digital twin developments. Among other things, the framework must enable the connection of the digital twin with system data on concrete platforms.

Openness and Sustainability. To deal with the evolution of the systems and their environment, digital twins must be open to the addition of new models or data as they become available. The research challenge here consists in developing digital twin architectures and frameworks that support such open environment and which provides advanced composition operators to enable the integration of new models and data while the system is running. In this research challenge, it is also important to consider the impact on current model management solutions which have to provide more dynamic

---

[8] `https://www.eclipse.org/modeling/emf/`

[9] `https://www.jetbrains.com/mps/`

features to deal with runtime aspects of the model [27]. How this may be realized for long-living systems running for several decades gives this challenge an extra twist.

Uncertainty. Research on uncertainty modeling has emerged in the MDE community over the last years. Some recent work allows supporting the explicit modeling of uncertainty in design models, e.g. [7]. However, these concepts have not been integrated in current runtime modeling approaches yet. One research challenge consists in integrating the concept of uncertainty with runtime environments and digital twins to better deal with variations in received data, errors, changing operational conditions, and human behaviour. The identification of components that contribute to uncertainty and to which level is another research challenge.

Design Space Exploration. Digital twins should enable the exploration of different versions and variants of the same system at the virtual level, the granularity of the models may be different depending on the tasks to be performed. While there are already several approaches to perform design space exploration by using search or simulation techniques, it is less clear if these approaches are applicable in current digital twin scenarios with respect to responsiveness and scalability. The research challenge is to efficiently perform the simulations in relation to the exploration in order to make informed decisions, specially if uncertainty is involved about future states of the system.

Inconsistency Management. If multiple models from different domains are involved in the creation of a digital twin, a certain level of consistency between the individual models is required, but at the same time, inconsistencies must be also acceptable and highlighted at certain times [17]. Hence, while model management and inter-model consistency approaches are crucial, the main challenge is to be able to integrate models that are created by means of other tools, e.g., technical drawings. However, model exchange and interoperability in tool chains is still a major issue [15, 32] and novel techniques may be required to process truly heterogeneous models.

Models AND Data. Finally, the complementarity and duality of models and data in the specific context of digital twins must be addressed, in order to perform, for example, model optimization based on data obtained from digital twins and real machines. Especially, the efficient representation of historical data in models would allow for new temporal reasoning capabilities based on temporal models [20].

5 Conclusion

In this paper, we outlined how the current state of the art in MDE that can contribute to main challenges in the domain of digital twins. For this, we identified three major contributions for engineering digital twins: model management approaches for dealing with heterogeneous models, models@runtime for synchronizing the digital twin with the actual system, and collaborative modeling based on model versioning systems.

Next to and based on these contributions, we also identified a number of research challenges related to the development of digital twins. The systems engineering community may re-use the techniques, tools and methodologies developed within the MDE community, instead of starting from scratch. At the same time, they have to be adapted to this new context in order to be useful for systems engineers in the particular fields.

References

1. G. Acampora, A. Vitiello, B. Di Stefano, W. M. P. v. d. Aalst, C. W. Günther, and H. M. W. Verbeek. IEEE 1849TM: The XES Standard: The Second IEEE Standard Sponsored by IEEE Computational Intelligence Society. *IEEE Computational Intelligence Magazine*, pages 4–8, May 2017.
2. L. Addazi, A. Cicchetti, J. D. Rocco, D. D. Ruscio, L. Iovino, and A. Pierantonio. Semantic-based model matching with emfcompare. In B. S. Tanja Mayerhofer, Alfonso Pierantonio and D. Tamzalit, editors, *10th Workshop on Models and Evolution*, pages 40–49. CEUR-WS, October 2016.
3. T. Allweyer. *BPMN 2.0*. BoD, 2010.
4. N. Bencomo. The role of models@run.time in autonomic systems: Keynote. In *Proceedings of the IEEE International Conference on Autonomic Computing (ICAC)*, pages 293–294, 2017.
5. N. Bencomo, A. Bennaceur, P. Grace, G. S. Blair, and V. Issarny. The role of models@run.time in supporting on-the-fly interoperability. *Computing*, 95(3):167–190, 2013.
6. N. Bencomo, S. Götz, and H. Song. Models@run.time: a guided tour of the state of the art and research challenges. *Software & Systems Modeling*, 2019.
7. M. F. Bertoa, N. Moreno, G. Barquero, L. Burgueño, J. Troya, and A. Vallecillo. Expressing measurement uncertainty in OCL/UML datatypes. In *Proceedings of the 14th European Conference on Modelling Foundations and Applications (ECMFA)*, pages 46–62, 2018.
8. G. Blair, N. Bencomo, and R. B. France. Models@ run. time. *Computer*, 42(10):22–27, 2009.
9. F. Bordeleau, B. Combemale, R. Eramo, M. van den Brand, and M. Wimmer. Tool-support of socio-technical coordination in the context of heterogeneous modeling. In *Proceedings of MODELS 2018 Workshops, co-located with ACM/IEEE 21st International Conference on Model Driven Engineering Languages and Systems (MODELS 2018)*, pages 423–425, 2018.
10. J. Borky and T. Bradley. *Effective Model-Based Systems Engineering*. Springer, 2019.
11. E. Bousse, T. Degueule, D. Vojtisek, T. Mayerhofer, J. DeAntoni, and B. Combemale. Execution framework of the GEMOC studio (tool demo). In T. van der Storm, E. Balland, and D. Varró, editors, *Proceedings of the 2016 ACM SIGPLAN International Conference on Software Language Engineering, Amsterdam, The Netherlands, October 31 - November 1, 2016*, pages 84–89. ACM, 2016.
12. P. Brosch, M. Seidl, K. Wieland, M. Wimmer, and P. Langer. We can work it out: Collaborative conflict resolution in model versioning. In *ECSCW 2009*, pages 207–214. Springer, 2009.
13. L. Burgueño, F. Ciccozzi, M. Famelis, G. Kappel, L. Lambers, S. Mosser, R. F. Paige, A. Pierantonio, A. Rensink, R. Salay, G. Taentzer, A. Vallecillo, and M. Wimmer. Contents for a model-based software engineering body of knowledge. *Software and Systems Modeling*, 18(6):3193–3205, 2019.
14. B. H. C. Cheng, B. Combemale, R. B. France, J.-M. Jézéquel, and B. Rumpe. *On the Globalization of Domain-Specific Languages*, pages 1–6. Springer, 2015.
15. B. Combemale, J. DeAntoni, B. Baudry, R. B. France, J. Jézéquel, and J. Gray. Globalizing modeling languages. *Computer*, 47(6):68–71, 2014.
16. C. Eastman, P. Teicholz, R. Sacks, and K. Liston. *BIM Handbook: A Guide to Building Information Modeling for Owners, Managers, Designers, Engineers and Contractors*. Wiley Publishing, 2008.
17. S. Feldmann, K. Kernschmidt, M. Wimmer, and B. Vogel-Heuser. Managing inter-model inconsistencies in model-based systems engineering: Application in automated production systems engineering. *J. Syst. Softw.*, 153:105–134, 2019.

18. S. Friedenthal, A. Moore, and R. Steiner. *A Practical Guide to SysML: Systems Modeling Language*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2008.
19. P. Fritzson and V. Engelson. Modelica — a unified object-oriented language for system modeling and simulation. In E. Jul, editor, *ECOOP'98 — Object-Oriented Programming*, pages 67–90, Berlin, Heidelberg, 1998. Springer Berlin Heidelberg.
20. A. Gómez, J. Cabot, and M. Wimmer. TemporalEMF: A Temporal Metamodeling Framework. In *Proceedings of the 37th International Conference on Conceptual Modeling (ER)*, pages 365–381, 2018.
21. J. Kienzle, G. Mussbacher, B. Combemale, L. Bastin, N. Bencomo, J. Bruel, C. Becker, S. Betz, R. Chitchyan, B. H. C. Cheng, S. Klingert, R. F. Paige, B. Penzenstadler, N. Seyff, E. Syriani, and C. C. Venters. Toward model-driven sustainability evaluation. *Commun. ACM*, 63(3):80–91, 2020.
22. P. Klint, T. van der Storm, and J. J. Vinju. EASY Meta-programming with Rascal. In J. M. Fernandes, R. Lämmel, J. Visser, and J. Saraiva, editors, *Generative and Transformational Techniques in Software Engineering III - GTTSE 2009*, volume 6491 of *LNCS*, pages 222–289. Springer, 2009.
23. D. Kolovos, L. Rose, R. Paige, and A. Garcia-Dominguez. *The Epsilon Book*. Eclipse, 2010.
24. D. S. Kolovos, D. Di Ruscio, A. Pierantonio, and R. F. Paige. Different models for model matching: An analysis of approaches to support model differencing. In *2009 ICSE Workshop on Comparison and Versioning of Software Models*, pages 1–6. IEEE, 2009.
25. H. Krahn, B. Rumpe, and S. Völkel. Monticore: Modular development of textual domain specific languages. In R. F. Paige and B. Meyer, editors, *Objects, Components, Models and Patterns, 46th Int. Conference, TOOLS EUROPE 2008*, volume 11 of *Lecture Notes in Business Information Processing*, pages 297–315. Springer, 2008.
26. 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.
27. A. Mazak and M. Wimmer. Towards liquid models: An evolutionary modeling approach. In *Proceedings of the 18th IEEE Conference on Business Informatics (CBI)*, pages 104–112, 2016.
28. N. Mohammadi and J. E. Taylor. Smart city digital twins. In *2017 IEEE Symposium Series on Computational Intelligence (SSCI)*, pages 1–5, 2017.
29. I. Nagy, L. G. Cleophas, M. van den Brand, L. Engelen, L. Raulea, and E. X. L. Mithun. VPDSL: A DSL for software in the loop simulations covering material flow. In *Proceedings of the 17th IEEE International Conference on Engineering of Complex Computer Systems (ICECCS)*, pages 318–327, 2012.
30. R. Piascik, J. Vickers, D. Lowry, S. Scotti, J. Stewart, and A. Calomino. Technology area 12: Materials, structures, mechanical systems, and manufacturing road map. Technical report, NASA Office of Chief Technologist, 2010.
31. J. Rumbaugh, I. Jacobson, and G. Booch. *Unified Modeling Language Reference Manual, The (2nd Edition)*. Pearson Higher Education, 2004.
32. W. Silva Torres, M. van den Brand, and A. Serebrenik. Model management tools for models of different domains: a systematic literature review. In *Proceedings of the 13th Annual IEEE International Systems Conference*, 2019.
33. A. Toulmé. Presentation of EMF compare utility. In *Eclipse Modeling Symposium*, 2006.
34. W. M. P. van der Aalst. *Process Mining - Data Science in Action, Second Edition*. Springer, 2016.
35. T. Vogel and H. Giese. Requirements and assessment of languages and frameworks for adaptation models. *CoRR*, abs/1805.08679, 2018.
36. S. Weekes. The rise of digital twins in smart cities. *Smart Cities World*, 2019.