Conceptual Modeling Foundations, Model-Driven Engineering and Digital Twins

Judith Michael
Software Engineering, RWTH Aachen University
Aachen, Germany
michael@se-rwth.de

***Abstract*—What is the relationship between conceptual models and other model types? And how can conceptual modeling contribute to the model-driven engineering of digital twins? This short paper aims to stimulate discussion between different research communities and show potential for conceptual modeling in applications.**

***Index Terms*—Conceptual Modeling, Model-Driven Engineering, Digital Twins, Model Types, Classification**

We aim to understand and handle the complexity of large software-intensive systems using software and systems modeling methods. Digital twins of such complex systems enable analyzing the past and present of a system, optimizing the running system, and predicting the future by using different models about and data of the twinned system [1]. Digital twin engineering and operation combines different model types [2], [3], e.g., for describing the physical asset [4], for generating the digital twin [5], for describing the runtime state and processes [6], [7], or linking the digital twin with other systems [8], [9]. This requires different disciplines to work together and understand the modeling methods representing different perspectives and the kinds of abstractions they make. This observation is underpinned by a study about conceptual modeling, business process modeling, and model-driven engineering communities [10]: About half of the community members agreed that they are closely connected and a large majority would like to be even more connected. However, this requires a better understanding of the modeling research approaches among them. To better ground our modeling research, it would be helpful to find answers to the following questions:

- How do conceptual models fit into different classifications of model types?
- What is the relationship between a conceptual model and models used in Model-Driven Engineering (MDE)?
- How can conceptual models contribute to digital twin engineering?
- How can conceptual modeling foundations contribute to the Model-Based Software Engineering Body of Knowledge?

In the following, we discuss these questions and give some first hints on where to start a community discussion.

Types of Models. Mayr and Thalheim [11] propose eight characteristics and their features for describing conceptual models. To improve the understanding between different communities, a comparison of the different types of models and their characteristics would help. According to the Model-Based Software Engineering Body of Knowledge [12], conceptual models are information models. Considering the argumentation of Mayr and Thalheim [11], they can be more than pure information models as they are enhanced by concepts from a concept space. Looking into the literature, there exist different model classifications, e.g., Boyes and Watson [13] differentiate between data models and physical entity models with several subtypes, Michael et al. [4] (based on Novak et al. [14]) show a classification of models in hydraulic engineering, e.g., conceptual, empirical, analytical, numerical, computational models. There exists a categorization based on time properties and focus of the models: *structure*, *behavior*, *function*. Other commonly used classifications differentiate between models used for *generation* or code synthesis, and *interpreted* models, e.g., models@runtime [15]. In addition, there are other terms for types of models we are using in software engineering, e.g., domain models, requirements engineering models, simulation models, and system models. It would be interesting to investigate which characteristics they share (e.g., by providing a table) and which characteristics differentiate them. In addition, we would need a Venn Diagram describing where they overlap (similar to the one used in [16] describing the relationship between low-code and MDE).

Conceptual Models and Model-Driven Engineering. In MDE, we often use the term *system models* to describe a set of models we use in a generation process, e.g., different types of UML models, OCL models, SysML, and GUI models in a textual representation. In Figure 1, we show a first approach for describing the overlap between system models and conceptual models. Here, the evolution of models over time is interesting. When (1) creating a conceptual model (Figure 1, what is needed to (2) evolve to be a conceptual and a system model, and what happens to this model type when, e.g., the notion space is not defined any more (3). What characteristics do a conceptual model need to be (I) a conceptual and a system model from its creation on, e.g., a clearly defined notion space? And if (A) a model is created as a system model, what does it need to identify additionally (B) as a conceptual model and in which cases could these additions be helpful?

In addition, the question occurs if conceptual models are per-se usable or not usable for MDE. And if they have to be

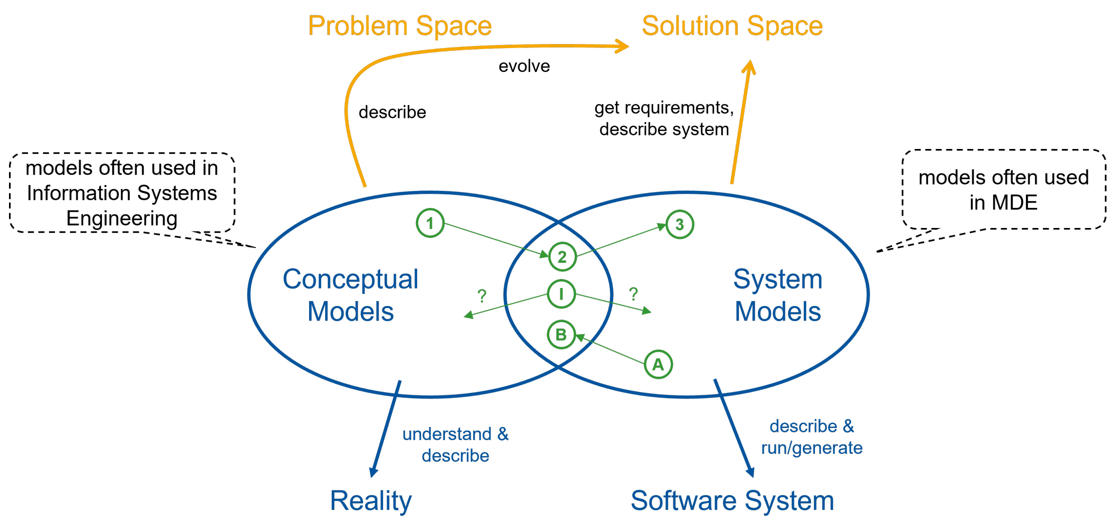

Fig. 1. Relationship between conceptual models and system models used for MDE

abstract and grounded in the problem space or have to be on a higher (conceptual) level so that they are not usable for MDE per definition.

Conceptual Models in Digital Twin Engineering. For digital twin engineering, it is becoming increasingly relevant to combine models from different application domains, with different levels of detail and characteristics, into a common system to better understand twinned objects, socio-technical systems [17], systems-of-systems [18], and ecosystems [19]. Digital twins use heterogeneous models, i.e. descriptive, prescriptive and prescriptive models [3], and their evolving data.

Up until now, the role of conceptual models in digital twin engineering has not been investigated well. There exist first works on using conceptual models for design thinking processes for digital twin engineering [20], or on the role of ontologies in digital twins [21]. A better foundation would contribute to a better understanding of what conceptual models can be used. Defining a notion space[1] is not only helpful but essential for any engineered digital twin, as this clarifies the used concepts and supports the understanding of different user groups. Thus, a better understanding of the influences of certain characteristics of conceptual models on the different participating users and application domains in every digital twin engineering process would be helpful.

Contributing to the Software Engineering Body of Knowledge. For a better integration of modeling theory for complex systems, we need to integrate the developed foundations of conceptual modeling in existing Body of Knowledge (BoK) collections, e.g., the Model-Based Software Engineering Body of Knowledge [12]. Here, it would be good to check the BoK and identify if additions from the conceptual modeling perspective would make sense. A Dagstuhl-like setting would be the right place to do so.

Conclusion. As identified in [10], collaborations between different modeling communities would be beneficial for various research topics. When looking at ideas and new visions of conceptual modeling [22] or recent discussions on abstractions engineering [23], we have to ask ourselves what the right place for conceptual modeling is. Coming from information system engineering and databases its methods seem to fit well in other research areas, however, its usage should not be just a relabeling of terms. We need to systematically specify in which development phase which kinds of systems with which representation paradigms and languages for which purpose conceptual models can be created for whom and what their usage characteristics are.

[1] Please note that the term *notion space* is used on purpose. It should not be mixed up with the term ontology. An ontology represents only one way to realize it. There are also other encyclopedic grounding options.

Acknowledgment

Funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) – Model-Based DevOps – 505496753. Website: https://mbdo.github.io.

References

[1] M. Heithoff, A. Hellwig, J. Michael, and B. Rumpe, "Digital Twins for Sustainable Software Systems," in *GREENS 2023*. IEEE, July 2023, pp. 19–23.

[2] B. Combemale, J.-M. Jézéquel, Q. Perez, D. Vojtisek, N. Jansen, J. Michael, F. Rademacher, B. Rumpe, A. Wortmann, and J. Zhang, "Model-Based DevOps: Foundations and Challenges," in *Int. Conf. on Model Driven Engineering Languages and Systems Companion (MODELS-C)*. ACM/IEEE, Oct 2023, pp. 429–433.

[3] R. Eramo, F. Bordeleau, B. Combemale, M. v. d. Brand, M. Wimmer, and A. Wortmann, "Conceptualizing digital twins," *IEEE Software*, vol. 39, no. 2, pp. 39–46, 2022.

[4] J. Michael, J. Blankenbach, J. Derksen, B. Finklenburg, R. Fuentes, T. Gries, S. Hendiani, S. Herlé, S. Hesseler, M. Kimm, J. C. Kirchhof, B. Rumpe, H. Schüttrumpf, and G. Walther, "Integrating models of civil structures in digital twins: State-of-the-Art and challenges," *Journal of Infrastructure Intelligence and Resilience*, 2024. [Online]. Available: https://doi.org/10.1016/j.iintel.2024.100100

[5] D. Bano, J. Michael, B. Rumpe, S. Varga, and M. Weske, "Process-Aware Digital Twin Cockpit Synthesis from Event Logs," *Journal of Computer Languages (COLA)*, vol. 70, 2022.

[6] N. Bencomo, S. Götz, and H. Song, "Models@run.time: a guided tour of the state of the art and research challenges," *Software and Systems Modeling*, vol. 18, no. 5, pp. 3049–3082, 2019.

[7] T. Brockhoff, M. Heithoff, I. Koren, J. Michael, J. Pfeiffer, B. Rumpe, M. S. Uysal, W. M. P. van der Aalst, and A. Wortmann, "Process Prediction with Digital Twins," in *Int. Conf. on Model Driven Engineering Languages and Systems Companion (MODELS-C)*. ACM/IEEE, October 2021, pp. 182–187.

[8] V. A. Shekhovtsov, S. Ranasinghe, H. C. Mayr, and J. Michael, "Domain Specific Models as System Links," in *Advances in Conceptual Modeling Workshops (ER'18)*. Springer, November 2018, pp. 330–340.

[9] J. C. Kirchhof, J. Michael, B. Rumpe, S. Varga, and A. Wortmann, "Model-driven Digital Twin Construction: Synthesizing the Integration of Cyber-Physical Systems with Their Information Systems," in *Proceedings of the 23rd ACM/IEEE International Conference on Model Driven Engineering Languages and Systems*. ACM, October 2020, pp. 90–101.

[10] J. Michael, D. Bork, M. Wimmer, and H. Mayr, "Quo Vadis Modeling? Findings of a Community Survey, an Ad-hoc Bibliometric Analysis, and Expert Interviews on Data, Process, and Software Modeling," *Journal Software and Systems Modeling (SoSyM)*, vol. 23, no. 1, pp. 7–28, Feb 2024. [Online]. Available: https://doi.org/10.1007/s10270-023-01128-y

[11] H. C. Mayr and B. Thalheim, "The triptych of conceptual modeling: A framework for a better understanding of conceptual modeling," *Softw. Syst. Model.*, vol. 20, no. 1, p. 7–24, feb 2021. [Online]. Available: https://doi.org/10.1007/s10270-020-00836-z

[12] 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*, vol. 18, no. 6, pp. 3193–3205, 2019.

[13] H. Boyes and T. Watson, "Digital twins: An analysis framework and open issues," *Computers in Industry*, vol. 143, p. 103763, 2022.

[14] P. Novak, V. Guinot, A. Jeffrey, and D. E. Reeve, *Hydraulic Modelling - an Introduction: Principles, methods, and applications*, 1st ed. London: Spon Press, 2010.

[15] N. Bencomo, S. Götz, and H. Song, "Models@run.time: a guided tour of the state of the art and research challenges," *Software and Systems Modeling*, vol. 18, no. 5, pp. 3049–3082, 2019.

[16] D. Di Ruscio, D. Kolovos, J. de Lara, A. Pierantonio, M. Tisi, and M. Wimmer, "Low-code development and model-driven engineering: Two sides of the same coin?" *Software and Systems Modeling*, 2022.

[17] F. Bonetti, A. Bucchiarone, J. Michael, A. Cicchetti, A. Marconi, and B. Rumpe, "Digital Twins of Socio-Technical Ecosystems to Drive Societal Change," in *Int. Conf. on Model Driven Engineering Languages and Systems Companion (MODELS-C)*. ACM/IEEE, 2024.

[18] J. Michael, J. Pfeiffer, B. Rumpe, and A. Wortmann, "Integration Challenges for Digital Twin Systems-of-Systems," in *10th IEEE/ACM International Workshop on Software Engineering for Systems-of-Systems and Software Ecosystems*. IEEE, 2022. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/9808817

[19] J. Michael, I. David, and D. Bork, "Digital Twin Evolution for Sustainable Smart Ecosystems," in *Int. Conf. on Model Driven Engineering Languages and Systems Companion (MODELS-C)*. ACM, 2024.

[20] A. Voelz, C. Muck, D. M. Amlashi, and D. Karagiannis, "Bridging haptic design thinking and cyber-physical environments through digital twins using conceptual modeling," in *22nd Int. Conf. on Perspectives in Business Informatics Research (BIR 2023)*, 2023, pp. 195–208.

[21] E. Karabulut, S. F. Pileggi, P. Groth, and V. Degeler, "Ontologies in digital twins: A systematic literature review," *Future Generation Computer Systems*, vol. 153, pp. 442–456, 2024.

[22] R. Lukyanenko, B. Samuel, V. Storey, and A. Sturm, "Conceptual modeling systems: A vision for the future of conceptual modeling," in *ER Forum and PhD Symposium 2022*, vol. 3211. CEUR-WS, 2022.

[23] N. Bencomo, J. Cabot, M. Chechik, B. Cheng, B. Combemale, A. Wasowski, and S. Zschaler, "Abstraction engineering," 2024. [Online]. Available: https://arxiv.org/abs/2408.14074