

Conceptualizing Digital Twins

Citation for published version (APA):

Eramo, R., Bordeleau, F., Combemale, B., van den Brand, M., Wimmer, M., & Wortmann, A. (2022). Conceptualizing Digital Twins. *IEEE Software*, 39(2), 39-46. <https://doi.org/10.1109/MS.2021.3130755>

Document license:

TAVERNE

DOI:

[10.1109/MS.2021.3130755](https://doi.org/10.1109/MS.2021.3130755)

Document status and date:

Published: 01/03/2022

Document Version:

Publisher's PDF, also known as Version of Record (includes final page, issue and volume numbers)

Please check the document version of this publication:

- A submitted manuscript is the version of the article upon submission and before peer-review. There can be important differences between the submitted version and the official published version of record. People interested in the research are advised to contact the author for the final version of the publication, or visit the DOI to the publisher's website.
- The final author version and the galley proof are versions of the publication after peer review.
- The final published version features the final layout of the paper including the volume, issue and page numbers.

[Link to publication](#)

General rights

Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

- Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
- You may not further distribute the material or use it for any profit-making activity or commercial gain
- You may freely distribute the URL identifying the publication in the public portal.

If the publication is distributed under the terms of Article 25fa of the Dutch Copyright Act, indicated by the "Taverne" license above, please follow below link for the End User Agreement:

www.tue.nl/taverne

Take down policy

If you believe that this document breaches copyright please contact us at:

openaccess@tue.nl

providing details and we will investigate your claim.

Conceptualizing Digital Twins

Romina Eramo, University of L'Aquila

Francis Bordeleau, École de technologie supérieure

Benoit Combemale, University of Rennes 1

Mark van den Brand, Eindhoven University of Technology

Manuel Wimmer, Johannes Kepler University Linz

Andreas Wortmann, University of Stuttgart

// Properly arranging models, data sources, and their relations to engineer digital twins is challenging. We 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 lifecycle. //



DIGITAL TWINS (DTS) are appearing everywhere, in agriculture, construction, engineering, production, and medicine, to mention just a few prominent examples. Research and industry have produced various definitions,^{1,2} ranging from underspecified (digital

replica or virtual counterpart), overly narrow (virtual representation based on augmented reality technology), to utopian (complete digital representation) approaches. The concepts found in the literature range between 1) high-fidelity design-time models used for design-space exploration, dimensioning, or validation, and 2) 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 lifecycle and, at the same time, can interact with and influence the AS.³ DTs can be built for a variety of purposes, such as the design, development, analysis, simulation, and operation of nondigital systems 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 (such as for design-space exploration) or at runtime (for example, 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 provide a set of services that use data, possibly obtained from the AS, and models for specific purposes with respect to the AS.⁴

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, their DTs also need to integrate various heterogeneous models to address the different aspects of the system. These models may be engineering models (for example, building information modeling, Modelica, Simulink, and mathematical equations) or software models (such as the Architecture Analysis and Design Language, MontiArc, Systems Modeling Language, Unified Modeling Language, or models in some kind of domain-specific language).

Digital Object Identifier 10.1109/MS.2021.3130755
Date of current version: 14 February 2022

To make sense of these models during a system’s design time and its run-time, 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^{1,2} 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.

A Conceptual Framework for DTs

We consider a DT to comprise a set of models and data associated with the AS that enables the creation of a set of services,⁴ corresponding to the functionalities provided by the DT. For instance, the DT can use simulation (even in communication with other DTs) to aid decision making for maintenance or improvement of the AS based on new requirements and/or available resources.

Figure 1 presents a conceptual framework for DTs described using the Models and Data (MODA) framework,⁵ a conceptual modeling⁶ framework that aims to support 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 artifacts on a conceptual level. In this way, we follow but also extend the viewpoints of existing frameworks proposed in the literature.^{1,2}

A DT is a virtual representation of an AS that exists within its environment. The AS produces data that are related to different aspects of the system, and the DT captures these 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.

The AS 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, which 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.² 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 dataflows (see the arrow labeled Monitoring in Figure 1): data obtained from monitoring and sensing, measured data, and external/historical data from

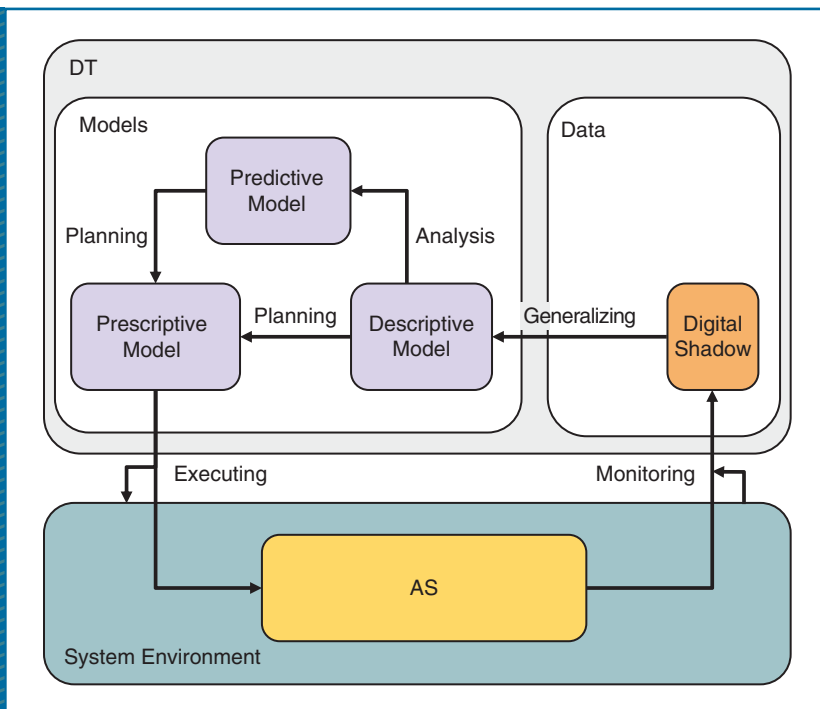


FIGURE 1. The conceptual framework for DTs based on MODA.⁵ MODA: Models and Data.

the system or its environment (for example, environmental data, technical data, and constraints). To this end, DTs comprise *digital shadows*, that is, purposefully abstracted and aggregated data structures⁴ that represent a one-way dataflow 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.¹ Models address different aspects and disciplines of a system (for example, engineering models, software models, and scientific models) and can be of various language types.³

As defined in the MODA framework,⁵ 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 AS, facilitating understanding, and enabling analysis.⁵ The arrow labeled Generalizing (between digital shadow and descriptive model) in Figure 1 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 behavior 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 (such as Petri nets), simulation (for example, Simulink), and machine learning (such as different types of neural networks).

By 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.⁵ The arrows labeled Analysis and Planning

represent decision support activities (for example, a what-if analysis) used 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.

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 lifecycle, notably design, manufacturing, maintenance, and utilization.⁷ In what follows, we characterize these applications using our conceptual framework. Figure 2 illustrates the set of DT applications as instances of the DT conceptual framework.

Design

The design of new systems requires the development team to collaborate with stakeholders by exchanging and sharing design data and relevant information⁷ 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 allow one to explore different design alternatives based on actual data.

In such cases, data are collected from both the system being designed and the environment (stakeholders). The DT can relate system functions and appearance, technical and process data, tests results from the design process, and stakeholder/customer feedback. These data are recorded and managed by the DT using a set of digital shadows, each focused on data related to different design aspects.

On the model side, these data, together with historical data of similar systems, are used to produce different types of descriptive models, such as feature, structure, behavior, and domain models. Besides allowing one to better understand different aspects of the system design, these 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 the accurate prediction of specific properties related to performance. Finally, decisions to improve the system design (for example, configuration, adaptation, and refactoring) are encoded in prescriptive models and then applied in the system design.

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 the automotive industry to help optimize different aspects of the manufacturing process, including production flow, product quality, resource management, waste reduction, and maintenance intervals.⁸

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, for example, lead time, percentage of productive time, and different quality metrics.

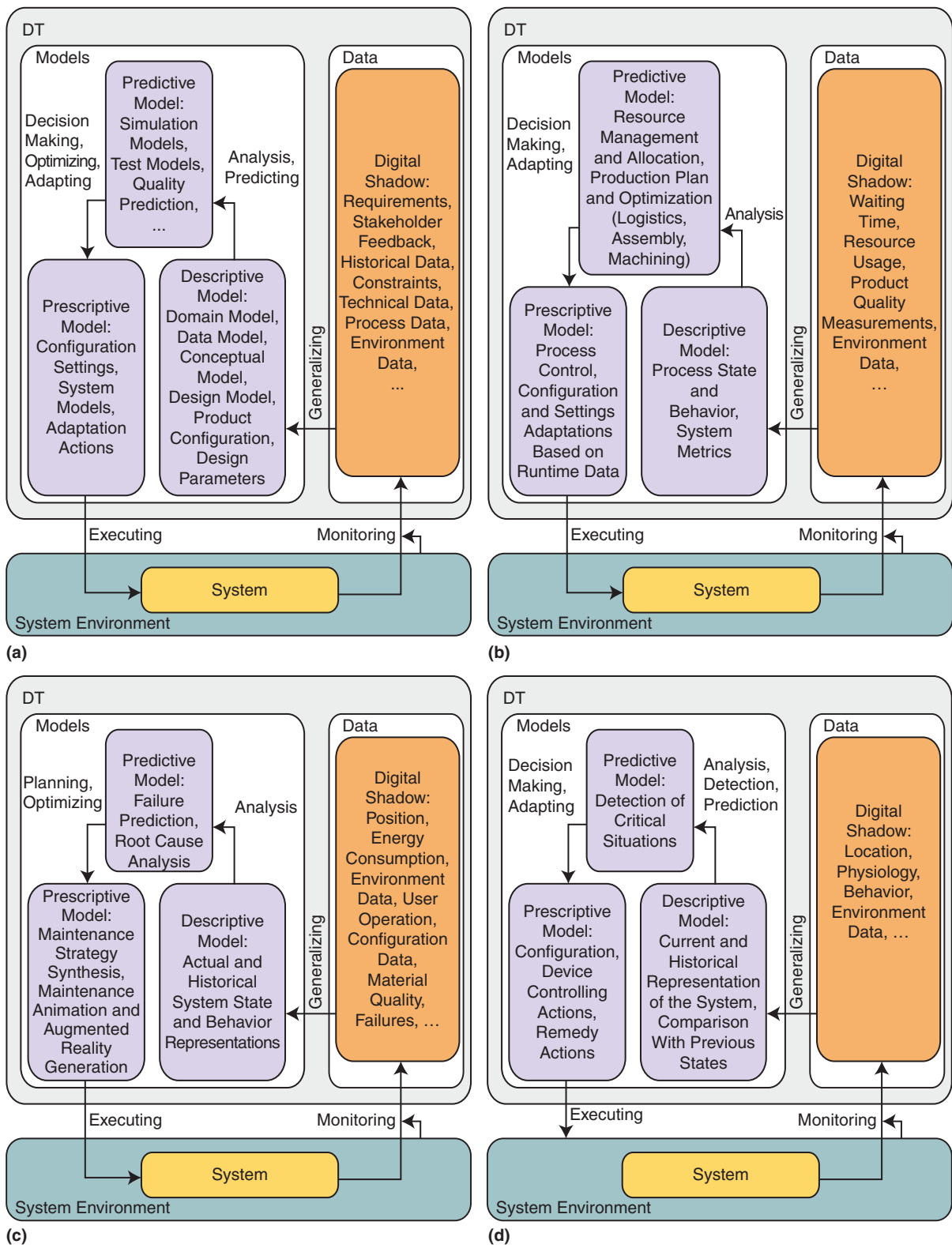


FIGURE 2. Instances of the DT conceptual framework representing different DT applications: (a) design, (b) manufacturing, (c) maintenance, and (d) utilization.

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 models⁹ and value stream maps.¹⁰ 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.

Maintenance

Complex systems (for example, aircraft, robots, automobiles, 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 one to move to the area of predictive maintenance.⁷

Real-time state data are transmitted to the digital shadow to realize the synchronous linkage between the AS 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, and failures, among others.

The descriptive model is built on top of the real-time data and the obtained historical data. By running relevant failure prediction algorithms, for example, based on machine learning

models, system failure prediction (the 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, for example, by applying artificial intelligence planning algorithms to also consider constraints from the product environment, such as currently processed orders.

Utilization

In addition to maintenance, the utilization of systems in a certain context is also currently a topic for DTs. For instance, DT technologies emerged in domains such as agriculture/farming¹¹ to improve/optimize natural, nonengineered systems. A main characteristic of these systems is that they cannot be directly controlled. The DT represents the system and its environment; it 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.¹²

Another example is a medical treatment system that can monitor varying health aspects of a patient to improve/optimize treatments for different health conditions.^{13,14} Concerning this application domain, several kinds of data are monitored and represented by the digital shadow, for example, physiology data retrieved from body sensors, location data retrieved from motion sensors, and behavioral data extracted by image analysis from cameras.²

Such digital shadows are then used to build the descriptive model, which 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, one 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. 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 countermeasures.

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. This description provides a language in which to discuss DTs, especially from a software engineering perspective. In addition, it allows the reuse of concepts across DTs because many DTs will be based on hybrid architectures, that is, a mix of the presented variants. The list of DT applications presented here is not exhaustive, but it demonstrates 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 monitor–analyze–plan–execute over a shared knowledge¹⁵ that also require 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 Bordeleau et al.,³ which opens up an important research line.

ABOUT THE AUTHORS



ROMINA ERAMO is an assistant professor at the University of L'Aquila, L'Aquila, 67100, Italy. Her research interests include model-driven engineering, software quality/engineering, DevOps, and digital twins. Eramo received her Ph.D. from the University of L'Aquila, Italy. Contact her at romina.eramo@univaq.it or <https://people.disim.univaq.it/romina.eramo/>.



BENOIT COMBEMALE is a full professor of software engineering at the University of Rennes 1, Rennes, 35065, France. He leads the Computer Science Department at the Rennes Engineering School ESIR, and is involved in the IRISA and Inria labs. His research interests include model-driven software and systems engineering, software language engineering, and software validation and verification. Combemale received his Ph.D. from the University of Toulouse, and his Habilitation from the University of Rennes. He is a Member of IEEE. Contact him at <http://combemale.fr> or benoit.combemale@irisa.fr.



FRANCIS BORDELEAU is a professor at École de technologie supérieure (ETS), Montreal, H3C 1K3, Canada. He currently holds the ETS Kaloom-Telus Industry Research Chair in DevOps. His research interests include model-based engineering, software processes, DevOps, and digital twins. Bordeleau received his Ph.D. in electrical engineering from Carleton University, Ottawa, Canada. He is a Member of IEEE. Contact him at <https://www.etsmtl.ca/en/research/professors/fbordeleau/> or francis.bordeleau@etsmtl.ca.



MARK VAN DEN BRAND is a full professor of software engineering and technology in the Department of Mathematics and Computer Science at Eindhoven University of Technology, NL-5612 AE, The Netherlands, and a visiting professor at Royal Holloway, University of London. His research interests include model-driven engineering, domain-specific languages, meta-modeling, and digital twins. van den Brand received his Ph.D. from the Radboud University in the Netherlands. He is on the editorial board of *Science of Computer Programming*, *Open Computer Science*, and *Computer Languages* and is deputy editor in chief of *The Journal of Object Technology*. Contact him at <https://www.tue.nl/en/research/researchers/mark-van-den-brand/> or m.g.j.v.d.brand@tue.nl.



MANUEL WIMMER is a full professor leading the Institute of Business Informatics-Software Engineering at Johannes Kepler University Linz, Linz, 4040, Austria, and the head of the Christian Doppler Laboratory for Model-Integrated Smart Production. His research interests comprise foundations of model engineering techniques in tool interoperability, legacy modeling tool modernization, model versioning and evolution, and industrial engineering. Wimmer received his Ph.D. from TU Wien, Austria. Contact him at <https://www.se.jku.at/manuel-wimmer/> or manuel.wimmer@jku.at.



ANDREAS WORTMANN is a professor at the University of Stuttgart, Stuttgart, 70174, Germany. His research interests include model-driven engineering, systems engineering, software language engineering, and robotics. Wortmann received his Ph.D. from RWTH Aachen 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 serves on the editorial boards of *The Journal of Object Technology* and *Software and Systems Modeling*. Contact him at www.wortmann.ac or andreas.wortmann@isw.uni-stuttgart.de.

More (literature) research needs to be done to identify other architectural patterns or combinations of these patterns that may be documented in a pattern catalog 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, as is already provided for simulators realizing the functional mock-up interface standard. Finally, the conceptual elements have to be mapped to concrete technologies, which would allow one to deploy DTs on dedicated platforms with a higher automation potential. 🌐

References

1. W. Kritzinger, M. Karner, G. Traar, J. Henjes, and W. Sihn, "Digital twin in manufacturing: A categorical literature review and classification," *IFAC-PapersOnLine*, vol. 51, no. 11, pp. 1016–1022, 2018, doi: 10.1016/j.ifacol.2018.08.474.
2. R. Minerva, G. M. Lee, and N. Crespi, "Digital twin in the IoT context: A survey on technical features, scenarios, and architectural models," *Proc. IEEE*, vol. 108, no. 10, pp. 1785–1824, 2020, doi: 10.1109/JPROC.2020.2998530.
3. 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. 1st Int. Conf. Syst. Model. Manage. (ICSMM)*, vol. 1262, pp. 43–54, Oct. 2020, doi: 10.1007/978-3-030-58167-1_4.
4. P. Bibow *et al.*, "Model-driven development of a digital twin for injection molding," in *Proc. 32nd Int. Conf. Adv. Inf. Syst. Eng. (CAiSE)*, vol. 12127, pp. 85–100, Jun. 2020, doi: 10.1007/978-3-030-49435-3_6.
5. B. Combemale *et al.*, "A hitchhiker's guide to model-driven engineering for data-centric systems," *IEEE Softw.*, vol. 38, no. 4, pp. 71–84, 2021, doi: 10.1109/MS.2020.2995125.
6. D. Schmidt, "Guest editor's introduction: Model-driven engineering," *Computer*, vol. 39, no. 2, pp. 25–31, 2006, doi: 10.1109/MC.2006.58.
7. F. Tao, J. Cheng, Q. Qi, M. Zhang, H. Zhang, and F. Sui, "Digital twin-driven product design, manufacturing and service with big data," *Int. J. Adv. Manuf. Technol.*, vol. 94, nos. 9–12, pp. 3563–3576, 2018, doi: 10.1007/s00170-017-0233-1.
8. 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*, vol. 5, no. 4, pp. 653–661, 2019, doi: 10.1016/j.eng.2019.01.014.
9. J. Recker, M. Rosemann, M. Indulska, and P. Green, "Business process modeling—a comparative analysis," *J. Assoc. Inf. Syst.*, vol. 10, no. 4, pp. 333–363, 2009, doi: 10.17705/1jais.00193.

10. P. Hines and N. Rich, "The seven value stream mapping tools," *Int. J. Operations Prod. Manage.*, vol. 17, no. 1, pp. 46–64, 1997, doi: 10.1108/01443579710157989.
11. S.-K. Jo, D.-H. Park, H. Park, and S.-H. Kim. "Smart livestock farms using digital twin: Feasibility study," in *Proc. Int. Conf. Inf. Commun. Technol. Convergence (ICTC)*, 2018, pp. 1461–1463, doi: 10.1109/ICTC.2018.8539516.
12. 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*, vol. 9, pp. 35,691–35,708, Feb. 2021, doi: 10.1109/ACCESS.2021.3061722.
13. H. Elayan, M. Aloqaily, and M. Guizani, "Digital twin for intelligent context-aware IoT healthcare systems," *IEEE Internet Things J.*, early access, 2021, doi: 10.1109/JIOT.2021.3103635.
14. Y. Liu *et al.*, "A novel cloud-based framework for the elderly health-care services using digital twin," *IEEE Access*, vol. 7, pp. 49,088–49,101, 2019, doi: 10.1109/ACCESS.2019.2909828.
15. J. Kephart and D. Chess, "The vision of autonomic computing," *Computer*, vol. 36, no. 1, pp. 41–50, 2003, doi: 10.1109/MC.2003.1160055.



IEEE Security & Privacy magazine provides articles with both a practical and research bent by the top thinkers in the field.

- stay current on the latest security tools and theories and gain invaluable practical and research knowledge,
- learn more about the latest techniques and cutting-edge technology, and
- discover case studies, tutorials, columns, and in-depth interviews and podcasts for the information security industry.



computer.org/security

