Integration Challenges for Digital Twin Systems-of-Systems

Judith Michael[1], Jérôme Pfeiffer[2], Bernhard Rumpe[1], Andreas Wortmann[2]
[1]Software Engineering, RWTH Aachen University, Germany
[2]Institute for Control Engineering of Machine Tools and Manufacturing Units (ISW), University of Stuttgart, Germany
{michael,rumpe}@se-rwth.de,{jerome.pfeiffer,andreas.wortmann}@isw.uni-stuttgart.de

ABSTRACT

Research and industry leverage digital twins to monitor and control (cyber-physical) systems in various domains. For their efficient engineering, these twins need to become Systems-of-Systems (SoS), in which digital twins of smaller systems (*e.g.,* a production machine) become parts of digital twins of larger systems (*e.g.,* a factory). Yet, research on digital twins as SoS largely ignores reusing digital twins in SoS. Based on our experience in engineering digital twins with experts from various domains related to production systems engineering, we present insights on the challenges of composing and integrating that need to be addressed for efficient engineering of digital twins as SoS. These insights may guide future research on engineering digital twins as well as practitioners considering the challenges in building and composing digital twin systems-of-systems.

KEYWORDS

Digital Twins, Composition, Model-Driven Software Engineering, Systems-of-Systems.

ACM Reference Format:
Judith Michael[1], Jérôme Pfeiffer[2], Bernhard Rumpe[1], Andreas Wortmann[2]. 2022. Integration Challenges for Digital Twin Systems-of-Systems. In *10th IEEE/ACM International Workshop on Software Engineering for Systems-of-Systems and Software Ecosystems (SESoS'22), May 16, 2022, Pittsburgh, PA, USA.* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3528229.3529384

1 MOTIVATION

Research and industry leverage digital twins (DTs) to monitor and control (cyber-physical) systems in various domains, including autonomous driving [9], biology [16], medicine [21], smart manufacturing [29], and many more. They promise the tremendous potential to reduce cost and time and to improve our understanding of the represented systems. The various DTs serve different purposes, including analysis [25], control [30], and behavior prediction [17], and they are used at different times relative to the represented system, *e.g.,* before it exists to explore its design space [23] or during its runtime to optimize its behavior [5]. Despite the plethora of definitions [14, 18, 22] there is little consensus about what a DT actually is. This also is reflected in many of the available definitions being (1) ambiguous, by deferring to another undefined term, such as a "virtual representation" [1], a "computable virtual abstraction" [28] , or a "a virtual projection of the industrial facility into the cloud" [31]; (2) narrow, by focusing on specific use cases, domains, or technologies, such as a "digital model of the real network environment" [12] or a "virtual representation based on AR-technology" [25]; or (3) utopian, due to all-encompassing aspirations, such as an "integrated virtual model of a real-world system containing all of its physical information" [26], a "complete digital representation" [24]. As DTs need to become Systems-of-Systems (SoS) (see Figure 1), in which twins of smaller systems (*e.g.,* a production machine) become parts of or communicate with twins of larger systems (*e.g.,* a production line or a factory), this lack of foundations severely hampers their composition and integration.

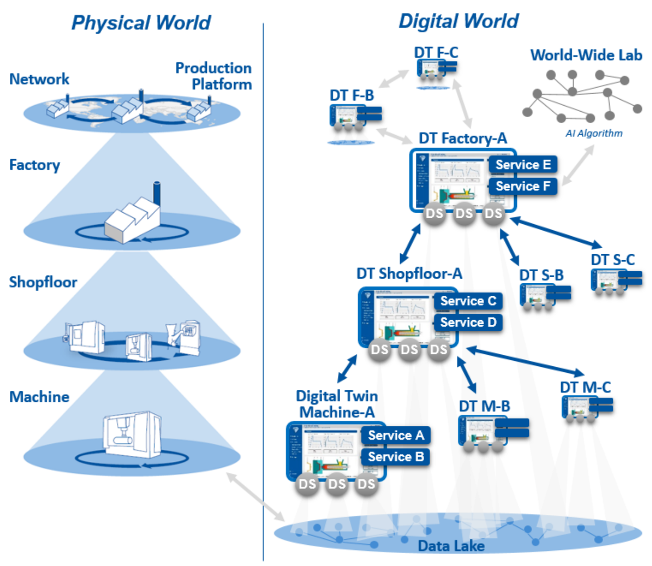

Figure 1: Digital twins for different production perspectives

To uncover the challenges in lifting DTs to SoS, we are conducting research on the foundations, engineering, and operations of DTs the interdisciplinary "Internet of Production"[1] (IoP) excellence cluster together with experts from artificial intelligence, automation, factory planning, human-machine-interaction, mechanical engineering, and software engineering. In this position paper, we

Funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Germany's Excellence Strategy – EXC 2023 Internet of Production - 390621612 and the Bundesministerium für Wirtschaft und Klimaschutz (BMWK, Federal Ministry for Economic Affairs and Climate Action) under grant no. 19S21002L.

[1]Internet of Production excellence cluster: https://www.iop.rwth-aachen.de

present insights from this research on the composition and integration challenges that need to be addressed for an efficient engineering of DTs as SoS. These insights may guide future research on engineering DTs and practitioners in building and composing DTs.

Next, Section 2 discusses foundations of our understanding of DTs and illustrates their use as SoS before Section 3 discusses integration challenges and Section 4 concludes.

2 DIGITAL TWIN SYSTEMS-OF-SYSTEMS

Research on DTs [4, 19] usually focuses on one of two extremes. Either contributions – often from automation engineering [11] or medicine [8] – understand DTs as highly-precise simulation models used at system design time, or – often from computer science – consider DTs as Internet of Things (IoT) systems capable of retrieving and operating on data from an observed (original) system. Sometimes, DTs also are confused with digital shadows [3]. Based on research in the IoP, we distinguish digital shadows from DTs [4]. Digital shadows [3] are data structures tailored to a specific purpose, *e.g.,* for monitoring device parameters for predictive maintenance or to facilitate strategic decision making. They are "shadows" as their contents and structure follow the original system under the illumination of their specific purpose, *i.e.,* a predictive maintenance shadow might comprise other information than the decision-making shadow, but both change with the original system. DTs [10] are systems themselves. They comprise data, models, and services to use the original system for a specific purpose [6], which might include the purposes of digital shadows but can extend to controlling the original system directly as well. Hence, when the system changes, the twin changes, and vice versa. To this end, they may operate on and comprise digital shadows (and more), but they are software systems.

Often, DTs represent original systems that become integrated into SoS at some point in time. For instance, the DT of a sensor, capable of performing predictive maintenance analysis, might be integrated into a production device capable of the same analysis while using the sensor's DT to this effect. The production device itself might be integrated into a production line for which a DT performs local production optimization based, among others, on the maintenance information received from the DTs of the individual devices. And the production device is integrated into a factory for which a DT makes strategic decisions about distributing production tasks based on the information received from the production line twins. Similarly, DTs of buildings [7] might be vertically composed into a DT a city [2] or DTs of cars might be integrated horizontally with another to communicate traffic decisions [20]. Ultimately, the horizontal integration and vertical composition of DTs yields DT SoS – and their composition and integration are subject to various challenges.

An interesting example to discuss different aspects of DTs and their integration is the production domain. Figure 1 shows one possible scenario on how to realize different aspects of the physical world in the digital one. In the physical world, we have machines on the bottom layer which are organized on shop floors within factories. In a connected company network, more than one factory exists which needs to exchange data and provide management views on more abstract levels.

When creating a digital representation of the physical world, DTs might be developed for various levels of the real world, e.g., separate DTs for different machines which are able to control these machines, DTs for complete shopfloors, factories, or production networks that can adapt processes as well as machines. As each DT needs to interact with different physical objects and follows different purposes, e.g. machine A needs to optimize the needed resources and machine B needs to reduce the fault rate, they might also use different services and need different data and models within digital shadows derived from the data lake. Moreover, there might exist DTs for different perspectives in production, e.g., DTs for the production process, for specific tools, or the produced products during their complete lifecycle from engineering to delivery and re- or upcycling. This heterogeneity leads to a variety of challenges.

3 CHALLENGES

One has to overcome several challenges to evolve from DTs of smaller systems to more complex SoS. Those are related to either the components and supported functionality of a DT, such as data, models, services, APIs, access rights, and views, or they are related to the context of a DT implementation such as implementation technologies, frameworks, or black-box systems.

1. Horizontal integration of digital twin parts: DTs at least need to represent their original system. These systems often feature multiple views on them, even on the same level within the SoS. For instance, a car DT may feature a driver's view, a maintenance view, an insurance view, and a producer's view. These comprise different models (e.g., mathematical, computational, artificial intelligence), services, and data, all of which need to be properly integrated into the overall DT. Currently, DTs are created ad-hoc, bottom-up in a piecemeal fashion, without any automation, systematic approaches to the integration of its heterogeneous parts, which makes (i) engineering DTs, (ii) integrating them as SoS by reusing them overly complicated.

2. Vertical composition of digital twins: Where the engineering of DTs could be fostered by composing the individual system into a larger, single SoS, similar conflicts between the comprised data, models, and services need to be resolved. For instance, (i) the data of the DT to be embedded might be recorded on different levels of abstraction or in incompatible granularity, (ii) its services and their behaviors might contradict the behaviors of the DT it should be embedded in, and (iii) the models by being incompatible. Where multiple DTs of an original system are supported, their composition to an SoS is subject to the same challenges, but with the twist of all DTs of the same original system aspiring to be the true DT for their respective (potentially overlapping) views.

3. Composition of DTs for different perspectives: The main focus of DTs are specific cyber-physical systems such as a machine or tool. In recent years, the concept of DTs is also transferred to humans, organizations, and decision processes. They might exist in parallel, e.g., DTs for the workers in the production process along with DTs for the machines and products. Integrating these different perspectives is challenging, as their existing services, visualizations, and data might have no explicitly defined connections.

4. Connection of independently developed systems to a system-of-system: How the integration of DTs is possible is also related to the accessibility of the implementation, e.g., the DT for a machine was developed in-house, the DT at the factory level uses defined applications, such as SAP HANA. If applications are only available as a black box, at least defined interfaces must exist for the integration. However, aspects such as necessary visualizations or rights management must then be realized in a merging implementation.

5. Different lifecycle representations of the original system: Integrating multiple DTs to an SoS, *e.g.,* integrating the production device DTs into a factory DT, requires (i) being aware of, and (ii) harmonizing the potentially different lifecycle phases that the DTs represent their original systems for. Some DTs represent their OS as-designed [13], some as-manufactured [24], some as-operated [27]. These different representations not only might use different data, services, and models (*e.g.,* as-designed often uses CAD and simulation models whereas as-operated often uses software models) but also focus on different aspects of the OS and require different connections to it (as-designed often connects to simulation models whereas as-operated often connects to the real OS). Considering their different representation phases, hence, is vital when lifting DTs to SoS.

6. Protection of intellectual property: DTs not only comprise intellectual property (IP) in form of the data and models about their OS but also in form of their behavior and services. While integrating DTs to SoS across company boundaries or when selling a systems DT together with the system, this IP must be protected as the DT otherwise might leak valuable information. Yet, to integrate DTs into meaningfully operating SoS, they need to provide some data, models, and services to the overall SoS. This conflict of goals has to be solved systematically to foster the (automated) integration of DTs into SoS.

7. Privacy aspects of data: DTs have to handle data and models of the original system. When integrating DTs to SoSs, DT engineers have to make clear in which DT data can be used on which aggregation level, especially if data of human workers is used in combination with the original system. The use of their data has to be restricted to specific purposes regarding the General Data Protection Regulation (GDPR). When integrating DTs it may be necessary to aggregate the data or use it for other purposes, which leads to the need to consider the use of the data in each DT and evaluate whether to implement anonymization or pseudonymization techniques.

8. Rights and roles in the integrated DT: When integrating DTs, existing rights and roles have to be evaluated for the resulting SoS. Approaches such as simple inheritance of rights to other levels of abstraction might not be sufficient, as new roles might occur, old roles might become obsolete and new rights for services that only exist on other abstraction levels might have to be introduced. This hinders a fully automatic integration of rights and roles as it requires an additional requirements analysis phase and individual realizations.

9. Composition of heterogeneous twin implementations: DTs provide various services related to the original system, which are implemented in manifold ways depending on the purpose of the service, coding expertise of the engineer, or toolchain used to implement the DT. For instance, when the DT is used for simulation purposes, it may be implemented using CAD, whereas when it has to analyze and act on occurring data, the service may be implemented using python together with a data interface to the original system. A domain expert who has no or limited coding expertise may employ model-driven techniques to describe his DT, whereas a software engineer may use more expressive general-purpose languages. For DTs of SoS to be fully integrated, their composition has to overcome these implementation differences by providing appropriate interfaces, or composition mechanisms.

10. Conflicting constraints and requirements: Services provided by DTs rely on constraints and requirements associated with their purpose to the original system. These constraints can be conflicting when composing the services in a DT of an SoS. For instance, DTs on the machine or shopfloor level have real-time constraints and act on raw data that result in requirements for data acquisition and its efficiency, use of real-time capable communication protocols with the physical asset, etc. In contrast, the DTs on the factory level act on an abstracted set of data that does not require real-time or a highly efficient way of gathering data. Instead, on this level, data privacy and quality are important. When designing a DT of SoS these layers have to share their constraints and requirements to identify and solve possible conflicts.

11. Hierarchical functional abstraction: DTs act on all levels of the modern production world (see Figure 1). The different levels of the physical world imply different levels of abstraction of functionality within the corresponding DTs. For instance, a DT at the machine level requires functionality that enables it to connect and fetch data from the CPS in real-time. In contrast, the DT of a factory does not require real-time capability but instead provides functionality for factory management. Thus, when composing these two DTs, it is important that despite both services being integrated, i.e., the management gets notified when the data connection is broken, that the functionality of the services is separated in the intended level of abstraction.

12. Composition of interfaces DT2DT and DT2CPS: Formerly independent DTs may use different communication interfaces with different communication strategies, e.g., push, pull, continuous, synchronous, or asynchronous, between (1) the DT to the cyber-physical system, (2) the DT to the Data Lake (3) or in terms of a DT of SoS, the DT to other DTs. When composing these DTs into a larger DT of SoS, these interfaces need to be composed and conflicts have to be resolved.

13. Interoperability of models and simulation environments: When realizing a DT and its services various modeling techniques and simulation environments are available. When integrating these twins and their services, also their models and simulation environments need to be interoperable to properly work together in a DT of SoS. Ideally, this works in a black-box fashion without modification. There are already approaches towards this challenge with co-simulation [15] using the functional mock-up interface.

14. Integration of graphical user interfaces: The integration of DTs makes it necessary to integrate graphical user interfaces and provide advanced views with zoom-in and zoom-out capabilities. This works well for the vertical integration of DTs on different levels of abstraction as their existing interfaces can be combined using according to navigation functionality. The horizontal integration of DTs on the same level of abstraction is more challenging, as a new graphical interface has to be developed to provide user-friendly visualization and interaction possibilities.

15. Heterogeneous technology-stack and different distribution patterns of DTs: When composing different DTs, their heterogeneous technology stack and different distribution patterns such as Fog, Cloud, or Edge implementations on the machine are a big challenge. This would require mappings and migration between these technological realizations. Another challenge here is the effects on performance, e.g., the interaction of the DT for a machine responses faster than in the cloud on the factory level. As time is an important factor for interacting with DTs, this migration might also result in losing real-time abilities.

4 CONCLUSION

For their efficient engineering, individual digital twin systems must become composable to systems-of-systems of interconnected digital twins. To realize the vision of efficiently engineering digital twin systems-of-systems requires overcoming these challenges. Based on our experiences, with interdisciplinary research on digital twins in the Internet of Production excellence cluster, addressing horizontal integration, vertical composition, and composition of digital twins from different perspectives will have greater impact on the efficient engineering of digital twins and should be pursued urgently.

REFERENCES

[1] Aitor Ardanza, Aitor Moreno, Álvaro Segura, Mikel de la Cruz, and Daniel Aguinaga. 2019. Sustainable and flexible industrial human machine interfaces to support adaptable applications in the Industry 4.0 paradigm. *International Journal of Production Research* 57 (2019), 4045–4059.

[2] Mark Austin, Parastoo Delgoshaei, Maria Coelho, and Mohammad Heidarinejad. 2020. Architecting smart city digital twins: Combined semantic model and machine learning approach. *Journal of Management in Engineering* 36, 4 (2020).

[3] Fabian Becker, Pascal Bibow, Manuela Dalibor, Aymen Gannouni, Viviane Hahn, Christian Hopmann, Matthias Jarke, Istvan Koren, Moritz Kröger, Johannes Lipp, Judith Maibaum, Judith Michael, Bernhard Rumpe, Patrick Sapel, Niklas Schäfer, Georg J. Schmitz, Günther Schuh, and Andreas Wortmann. 2021. A Conceptual Model for Digital Shadows in Industry and its Application. In *Conceptual Modeling, ER 2021*. Springer, 271–281.

[4] Pascal Bibow, Manuela Dalibor, Christian Hopmann, Ben Mainz, Bernhard Rumpe, David Schmalzing, Mauritius Schmitz, and Andreas Wortmann. 2020. Model-Driven Development of a Digital Twin for Injection Molding. In *Int. Conf. on Advanced Information Systems Eng. (CAiSE'20) (LNCS, Vol. 12127)*. Springer.

[5] Florian Biesinger, Davis Meike, Benedikt Kraß, and Michael Weyrich. 2018. A Case Study for a Digital Twin of Body-in-White Production Systems General Concept for Automated Updating of Planning Projects in the Digital Factory. In *23rd Int. Conf. on Emerging Technologies and Factory Automation (ETFA)*.

[6] Tim Bolender, Gereon Bürvenich, Manuela Dalibor, Bernhard Rumpe, and Andreas Wortmann. 2021. Self-Adaptive Manufacturing with Digital Twins. In *2021 International Symposium on Software Engineering for Adaptive and Self-Managing Systems (SEAMS)*. IEEE Computer Society, 156–166.

[7] Alessandro Carbonari, Leonardo Messi, Berardo Naticchia, Massimo Vaccarini, and Massimiliano Pirani. 2020. Development of a BIM-based holonic system for real-time monitoring of building operational efficiency. *Frontiers of Engineering Management* 7, 1 (2020), 89–103.

[8] Neeraj Kavan Chakshu, Jason Carson, Igor Sazonov, and Perumal Nithiarasu. 2019. A semi-active human digital twin model for detecting severity of carotid stenoses from head vibration—A coupled computational mechanics and computer vision method. *Int. Journal for Numerical Methods in Biomedical Eng.* 35, 5 (2019).

[9] Ximing Chen, Eunsuk Kang, Shinichi Shiraishi, Victor M Preciado, and Zhihao Jiang. 2018. Digital behavioral twins for safe connected cars. In *21th ACM/IEEE Int. Conf. on Model Driven Engineering Languages and Systems*.

[10] Manuela Dalibor, Judith Michael, Bernhard Rumpe, Simon Varga, and Andreas Wortmann. 2020. Towards a Model-Driven Architecture for Interactive Digital Twin Cockpits. In *Conceptual Modeling*. Springer, 377–387.

[11] Violeta Damjanovic-Behrendt and Wernher Behrendt. 2019. An open source approach to the design and implementation of Digital Twins for Smart Manufacturing. *Int. Journal of Computer Integrated Manufacturing* 32, 4-5 (2019).

[12] Rui Dong, Changyang She, Wibowo Hardjawana, Yonghui Li, and Branka Vucetic. 2019. Deep learning for hybrid 5G services in mobile edge computing systems: Learn from a digital twin. *IEEE Trans.s on Wireless Comm.* 18, 10 (2019).

[13] Darina Dupláková, Marián Flimel, Ján Duplák, Michal Hatala, Svetlana Radchenko, and František Botko. 2019. Ergonomic rationalization of lighting in the working environment. Part I.: Proposal of rationalization algorithm for lighting redesign. *Int. Journal of Industrial Ergonomics* 71 (2019), 92–102.

[14] Marlene Eisenträger, Simon Adler, Matthias Kennel, and Sebastian Möser. 2018. Changeability in Engineering. In *IEEE Int. Conf. on Engineering, Technology and Innovation (ICE/ITMC)*.

[15] Cláudio Gomes, Bart Meyers, Joachim Denil, Casper Thule, Kenneth Lausdahl, Hans Vangheluwe, and Paul De Meulenaere. 2019. Semantic adaptation for FMI co-simulation with hierarchical simulators. *Simulation* 95, 3 (2019), 241–269.

[16] Matthew Joordens and Mo Jamshidi. 2018. On The Development of Robot Fish Swarms in Virtual Reality with Digital Twins. In *13th Annual Conference on System of Systems Engineering (SoSE)*.

[17] GL Knapp, Tuhin Mukherjee, JS Zuback, HL Wei, TA Palmer, Amitava De, and TJAM DebRoy. 2017. Building blocks for a digital twin of additive manufacturing. *Acta Materialia* 135 (2017), 390–399.

[18] Dmitry Kostenko, Nikita Kudryashov, Michael Maystrishin, Vadim Onufriev, Vyacheslav Potekhin, and Alexey Vasiliev. 2018. Digital Twin Applications: Diagnostics, Optimization and Prediction. *Annals of DAAAM & Proc.* 29 (2018).

[19] Werner Kritzinger, Matthias Karner, Georg Traar, Jan Henjes, and Wilfried Sihn. 2018. Digital Twin in manufacturing: A categorical literature review and classification. *IFAC-PapersOnLine* 51, 11 (2018), 1016–1022.

[20] Sathish AP Kumar, R Madhumathi, Pethuru Raj Chelliah, Lei Tao, and Shangguang Wang. 2018. A novel digital twin-centric approach for driver intention prediction and traffic congestion avoidance. *Journal of Reliable Intelligent Env.* 4, 4 (2018).

[21] Nathan Lauzeral, Domenico Borzacchiello, Michael Kugler, Daniel George, Yves Rémond, Alexandre Hostettler, and Francisco Chinesta. 2019. A model order reduction approach to create patient-specific mechanical models of human liver in computational medicine applications. *Computer methods and programs in biomedicine* 170 (2019), 95–106.

[22] Jinfeng Liu, Honggen Zhou, Xiaojun Liu, Guizhong Tian, Mingfang Wu, Liping Cao, and Wei Wang. 2019. Dynamic Evaluation Method of Machining Process Planning Based on Digital Twin. *IEEE Access* (2019).

[23] Eric Lutters. 2018. Pilot production environments driven by digital twins. *South African journal of industrial engineering* 29 (2018), 40–53.

[24] Claudio Mandolla, Antonio Messeni Petruzzelli, Gianluca Percoco, and Andrea Urbinati. 2019. Building a digital twin for additive manufacturing through the exploitation of blockchain: A case analysis of the aircraft industry. *Computers in Industry* 109 (2019), 134–152.

[25] Hergen Pargmann, Dörthe Euhausen, and Robin Faber. 2018. Intelligent big data processing for wind farm monitoring and analysis based on cloud-technologies and digital twins: A quantitative approach. In *2018 IEEE 3rd International Conference on Cloud Computing and Big Data Analysis (ICCCBDA)*.

[26] Kyu Tae Park, Young Wook Nam, Hyeon Seung Lee, Sung Ju Im, Sang Do Noh, Ji Yeon Son, and Hyun Kim. 2019. Design and implementation of a digital twin application for a connected micro smart factory. *International Journal of Computer Integrated Manufacturing* 32 (2019).

[27] Banavara R Seshadri and Thiagarajan Krishnamurthy. 2017. Structural health management of damaged aircraft structures using digital twin concept. In *25th AIAA/AHS Adaptive Structures Conference*.

[28] AMM Sharif Ullah. 2019. Modeling and simulation of complex manufacturing phenomena using sensor signals from the perspective of Industry 4.0. *Advanced Engineering Informatics* 39 (2019), 1–13.

[29] Jumyung Um, Jens Popper, and Martin Ruskowski. 2018. Modular augmented reality platform for smart operator in production environment. In *2018 IEEE Industrial Cyber-Physical Systems (ICPS)*.

[30] Igor Verner, Dan Cuperman, Amy Fang, Michael Reitman, Tal Romm, and Gali Balikin. 2018. Robot Online Learning Through Digital Twin Experiments: A Weightlifting Project. In *Online Engineering & Internet of Things*.

[31] NR Yusupbekov, FR Abdurasulov, FT Adilov, and AI Ivanyan. 2018. Application of Cloud Technologies for Optimization of Complex Processes of Industrial Enterprises. In *Int. Conf. on Theory and Appl. of Fuzzy Systems and Soft Computing*.