On the Challenges of Integrating Digital Twins

Benoit Combemale*, Jörg Kienzle†, Gunter Mussbacher‡, Pascal Archambault§, Jean-Michel Bruel¶, Lola Burgueño†, Betty H.C. Cheng‖, Loek Cleophas**, Gregor Engels††, Damien Foures‡‡, Stefan Klikovits[x], Vinay Kulkarni[xi], Judith Michael[xii], Sebastien Mosser[xiii], Houari Sahraoui§, Eugene Syriani§, Andreas Wortmann[xiv]

* *Université de Rennes, IRISA, Inria*, Email: benoit.combemale@irisa.fr

† *ITIS Software, Universidad de Málaga*, Email: {joerg.kienzle, lolaburgueno}@uma.es

‡ *McGill University*, Email: gunter.mussbacher@mcgill.ca

§ *DIRO, Université de Montréal*, Email: pascal.archambault@umontreal.ca, {sahraouh, syriani}@iro.umontreal.ca

¶ *IRIT, Université de Toulouse*, Email: jean-michel.bruel@irit.fr

‖ *Michigan State University*, Email: chengb@msu.edu

** *Eindhoven University of Technology*, Email: l.g.w.a.cleophas@tue.nl

†† *Paderborn University*, Email: engels@uni-paderborn.de

‡‡ *DDMS, Airbus Group*, Email: damien.da.foures@airbus.com

[x] *Johannes Kepler University*, Email: stefan.klikovits@jku.at

[xi] *Tata Consultancy Services Research*, Email: vinay.vkulkarni@tcs.com

[xii] *RWTH Aachen University*, Email: michael@se-rwth.de

[xiii] *McSCert, McMaster University*, Email: mossers@mcmaster.ca

[xiv] *Stuttgart University*, Email: andreas.wortmann@isw.uni-stuttgart.de

***Abstract*—Digital Twins (DTs) are a key technology for smart ecosystems to provide accurate digital representation of their constituents, e.g., smart buildings, farms, transportation, and citizens, as well as synchronization between the digital and the real subject, and the exploration of what-if scenarios and trade-off reasoning. To cope with emerging complex socio-technical ecosystems, we need to bring DTs together, which is a challenging endeavor. After giving a historical overview of system adaptation, we review the many enabling technologies that can help with DT integration. Using a smart city as an illuminating example to highlight scenarios that require integration of DTs, we discuss a model-based conceptual framework that identifies DT integration strategies and elaborate on nine key integration challenges that still need to be addressed. We call on the DT community to investigate these challenges.**

***Index Terms*—digital twin, integration, challenges**

I. Introduction

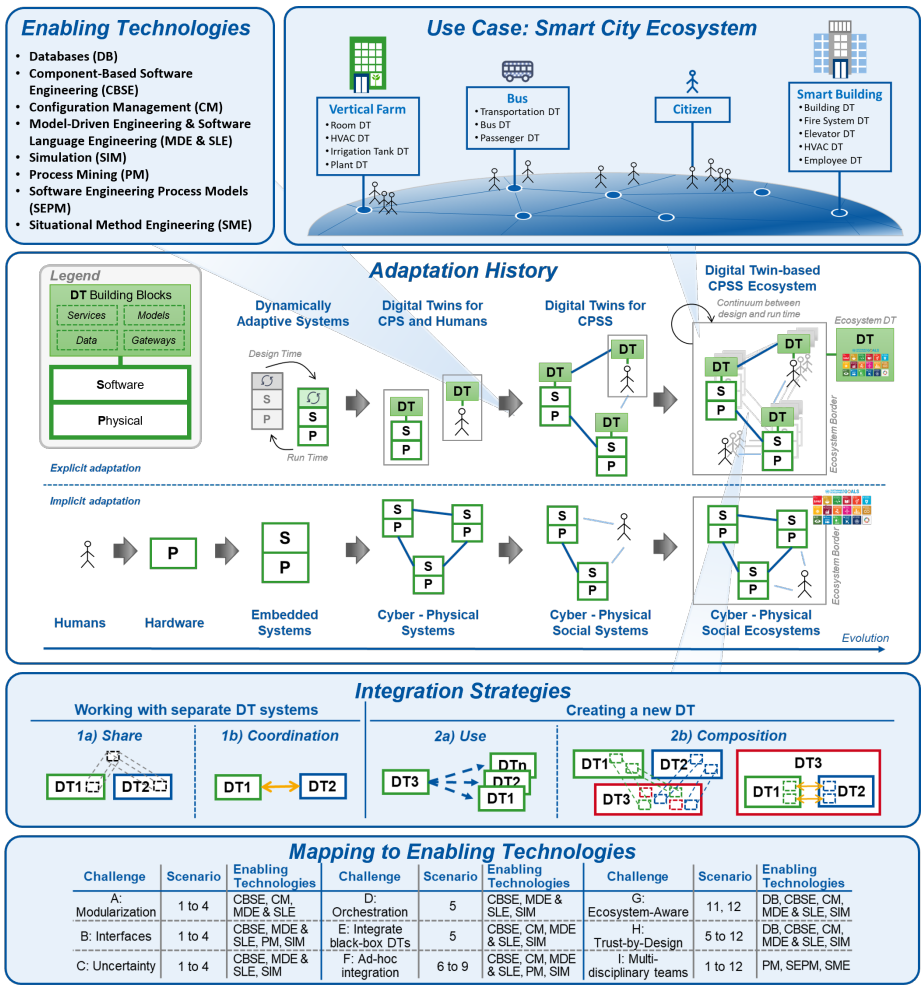
Digital twins (DTs) have emerged as a successful, modular means to interact with the physical world in a feedback loop involving the exploration of what-if scenarios at different abstraction levels [1], [2]. In the same way that humans do not live in isolation, but organize themselves into societies based on common goals, we need to bring digital twins together to support the analysis and optimization of emerging socio-technical ecosystems at a larger scale. Existing attempts have led to the proliferation of ad hoc solutions that integrate multiple DTs to reason about smart ecosystems [3].

In this vision paper based on a one-week workshop attended by all authors, we argue that a more systematic, integration-focused, model-based DT engineering approach is needed. Our key contribution is a model-based conceptual framework that identifies DT integration strategies for newly developed and existing DTs, DT integration challenges, relevant enabling technologies, and a mapping from integration challenges to enabling technologies motivated by real-world DT integration use cases. We give a historical overview of system adaptation to further motivate the framework. Furthermore, we describe the aggregate elements of a DT ecosystem and their design- and run-time relationships and provide a roadmap for DT developers comprising the detailed descriptions of nine technical and socio-technical challenges to DT integration (A–I).

II. Historical Overview

How did we get here? Society is greatly influenced by a transformative shift from engineered physical systems to software-intensive systems (Figure 1, middle), driven by the need for adaptability, cost-efficiency, and rapid innovation. By enabling quicker and possibly remote updates, rapid prototyping, and global distribution, software systems offer far greater flexibility than their physical counterparts. This facilitates a dynamic and interconnected technological landscape that fosters innovation and addresses the challenges of a rapidly evolving world.

Over time, this led to a stepwise extension of system types from embedded systems to distributed and integrated Cyber-Physical Systems (CPSs) (Figure 1, middle left-to-right). Once the role of humans is also considered, we call such systems Cyber-Physical Social Systems (CPSSs) or socio-technical systems [4]. These systems are deployed in specific domains, such as smart cities or smart manufacturing, ending up in smart ecosystems, where constituents interact and pursue shared objectives. Examples of such objectives are those in line with the 17 United Nations Sustainable Development Goals [5], such



Mapping to Enabling Technologies

Challenge	Scenario	Enabling Technologies	Challenge	Scenario	Enabling Technologies	Challenge	Scenario	Enabling Technologies
A: Modularization	1 to 4	CBSE, CM, MDE & SLE	D: Orchestration	5	CBSE, MDE & SLE, SIM	G: Ecosystem-Aware	11, 12	DB, CBSE, CM, MDE & SLE, SIM
B: Interfaces	1 to 4	CBSE, MDE & SLE, PM, SIM	E: Integrate black-box DTs	5	CBSE, CM, MDE & SLE, SIM	H: Trust-by-Design	5 to 12	DB, CBSE, CM, MDE & SLE, SIM
C: Uncertainty	1 to 4	CBSE, MDE & SLE, SIM	F: Ad-hoc integration	6 to 9	CBSE, CM, MDE & SLE, PM, SIM	I: Multi-disciplinary teams	1 to 12	PM, SEPM, SME

Fig. 1. Enabling Technologies for DT Systems (top left), Example Ecosystem with DTs (top right), A Brief History of Adaptation (middle 1st), Integration Strategies for DTs (middle 2nd), Mapping of Challenges to Enabling Technologies (bottom)

as responsible consumption and production, sustainability, as well as legal and ethical goals. In such ecosystems, data analytics and AI techniques are currently used to analyze and monitor their development [6].

In parallel, novel techniques have emerged where adaptivity management based on monitoring, analyzing, and planning of system updates became an explicit constituent of such a system. This started with Self-Adaptive Systems (SASs) [7] focusing on dynamic adaptation to changing conditions and uncertainty made available by an explicit feedback loop (e.g., MAPE-K [8]) (Figure 1, middle). SASs bring optimization, resilience, and fault tolerance. Together with advanced decision-making mechanisms, this makes them well-suited for coping with complex and dynamic environments, ensuring continuous evolution and improved system performance.

Further advancing this evolution is the detachment of the high-level control and planning part of the system into so-called DTs [9] on top of the functional and low-level control software and the physical system. A DT is a software system that purposefully represents an original, accurately reflects its changes, can act upon its original, and provides added value to users or other systems [10]. It manages models of and data from real-world entities, can perform analytics, and acts on the real entities to improve specific Key Performance Indicators (KPIs), such as sustainability, resilience, and resource efficiency [1]. Enhancing CPSSs with DTs enables deeper analysis, predictive modeling, and simulation of various scenarios. DTs also provide added value for better decision-making and performance optimization, thereby creating a bridge between the physical and digital realms [11]. If designed properly, they furthermore improve the modularity of SASs with a clear separation of concerns, better flexibility of the feedback loop, and advanced uncertainty management.

DT Integration. Engineering these DT-based CPSS ecosystems is a complex endeavor, which needs a deep understanding of how to integrate the building blocks of DTs. We assume a modular structure within a DT based on (i) *gateways* between the physical and the virtual worlds, (ii) *data* stores (a.k.a. digital shadows), (iii) descriptive, predictive, and prescriptive *models* [12], (iv) and atomic as well as composed *services* [9]. By integrating DTs, organizations gain flexibility and can better understand the entire ecosystem. In addition, it fosters

collaboration, ensures interoperability, and facilitates synchronized adaptation across multiple interconnected entities. Thus, we need novel standardized interface formats for those multimodal DT interfaces; novel composition (possibly evolving dynamically over time) and matching techniques for DT APIs; and an understanding of the properties of interacting DTs. This also leads to the demand for new holistic engineering processes to support software engineers in building and integrating DT-based CPSS ecosystems, where overall system properties are understood and can be handled.

After reviewing the state-of-the-art and existing enabling technologies, we use the example of a smart city (Figure 1, top) to present six use cases highlighting situations that require the integration of DTs, and identify open challenges for both practitioners and researchers relevant to the field of DT.

III. State-of-the-Art, Integration Strategies, and Enabling Technologies

In their discussion on DT challenges, Michael et al. [13], while not covering integration strategies, state that DT integration is challenging and not solved. Similarly, based on a systematic literature review, Olsson and Axelsson [14] argue that the challenges for systems-of-systems (SoS) also apply for systems of DTs, and that the implementation process for DTs is not yet fully understood. Cavalcante et al. [15] further investigate the interplay between DTs and SoS and related challenges but do not focus on DT integration. Bucaioni et al. [16] propose an MDE conceptual framework for the continuous engineering of federated DTs, along with challenges. For the use of DTs for civil structures, Michael et al. [17] describe applicable modeling approaches and discuss modeling challenges for this industry.

DT development relies on a broad and continuously growing foundation of technologies that over decades resulted in a multitude of enabling technologies [3], which, following the Digital Twin Capabilities Periodic Table [18], can be grouped along six functional dimensions[1]: data services, integration, intelligence, user experience (UX), management, and trustworthiness. Specific solutions exist for most of these capabilities.

Data services, for instance, include data ingestion, aggregation, and storage. For this, cloud-based solutions, such as AWS IoT TwinMaker [20] and Azure Digital Twins [21], or industrial on-premise technologies, such as OPC UA [22], address the need for such data management, but are lacking in other aspects, such as the DTs' behavioral description.

To represent the twinned system and give meaning, data must be linked to models, ensuring the complementarity and combination of inductive and deductive reasoning [23]. These models then have to be aligned, merged and/or orchestrated, synchronized, and overall managed systematically to be effectively used by the twins. For some of these challenges, solutions already exist, e.g., the MODA framework [12], MDE techniques (e.g. model transformations, model evolution), the Asset Administration Shell [22], and domain-specific languages [24]. These solutions should be leveraged, and possibly extended, for use in the DT context. Other enabling technologies have been already identified in the literature to cover most of these capabilities [25], [26]. Moreover, Martinelli et al. [27] and Visser et al. [28] present a hierarchical architecture for DTs in the manufacturing and automative industries, respectively, but do not describe how DT integration is to be accomplished. Kuruppuarachchi et al. [29] present yet another architecture for DTs. Schroeder et al. [30] discuss six topologies for DTs, including the aggregated DT. Onaji et al. [31] discuss DT integration but focus in particular on product and process DTs. Gil et al. propose an MDE approach [32] for the composition of DTs including co-simulation [33] but do not take the composition of inductive models into account. Gill et al. [34] report on initial work towards automated DT composition. Fu et al. [35] present a combined multi-view and multi-level approach for increased DT interoperability at the model level. Michael et al. [36] argue that MDE can help support the DT life cycle. Khedr and Fitzgerald [37] perform a systematic literature review, focusing on DT validation & verification, but also provide a coarse classification of DT composition approaches and state that uncertainty, complexity, and lack of standards are key technical challenges. Varela et al. [38] report on DT interoperability based on a systematic mapping study, highlighting ontologies and standardization as crucial solutions. Li et al. [39] also use ontologies to support DT integration. Pias et al. [40] focus on privacy and ethical issues in the context of interconnected DTs.

This paper focuses on the integration of DTs through a conceptual framework that identifies integration strategies. At a high level of abstraction, different integration strategies exist to support various scenarios. For example, DTs that evolve in the same context can share common building blocks, or they can coordinate using their interfaces (Strategies 1a and 1b in Figure 1, middle). Another strategy is to create a new DT that uses a set of existing DTs through their interfaces (Strategy 2a), or even build a new DT either by merging or composing their building blocks (Strategy 2b.1) or coordinating their building blocks (Strategy 2b.2).

Strategy 1a has the lowest complexity, as the DTs do not have to know about each other. Two DTs may share a building block but are otherwise independent of each other, requiring only some kind of agreement about the shared building block (e.g., same data structure, interfaces, etc). Strategy 2a has a similar complexity, as the used DTs (DT 1 to DTn in Figure 1) remain the same and DT3 only needs access to them via interfaces. Again, agreements are needed on the stability of the interfaces of existing DTs over time, similarly to Strategy 1a. Strategy 1b has a higher complexity in the integration process, as an agreement needs to reached about how building blocks are coordinated and how they relate to each other, especially in case of changes. Realizing Strategy 2b has the highest complexity, as each building block has to be checked for its ability and particular functionality to be reused. This might also mean, that building blocks are combined, e.g., gateways, models, or data are merged or certain services are connected

[1]A classification of such capabilities [19] exists but is organized according to main DT components

posing additional requirements on their validity restrictions and input/output specifications.

DT integration demands common, multi-modal, interfaces between DTs that provide access to their building blocks (gateways, data, models, and services). A plethora of existing techniques that enable interface-based modularization and composition exist in different areas of Computer Science, which should be used as a source of inspiration for enabling the DT integration (Figure 1, top left). This includes techniques from *Databases* (schema integration, federated databases, International Data Spaces, domain-specific repositories), *Component-Based Software Engineering* (the notion of component and the modularity it provides, provision and expectation of interfaces, API matching, ontologies and matching, coupling of components, service orchestration), *Configuration Management* (variants, variability and versioning, adaptation), *Model-Driven Engineering* and *Software Language Engineering* (metamodeling, model types/composition/versioning), and finally integration techniques from *Simulation*, *Co-simulation* and *"What If"-Analyses*. To cover the evolution of integrated DTs over time within an ecosystem, inspiration can be taken from areas such as *Process Mining* (process discovery, process conformance analysis, process prediction), *Software Engineering Process Models* (security-by-design, ethics-by-design), and *Situational Method Engineering* (iterative, incremental, agile).

The scientific community has longstanding experience in the development of approaches and solutions for the synergistic integration of these technologies. Given the explicit need for modeling capabilities within each DT, the extension of orchestration, synchronization, and merging of (meta-)models [41] for their use by DT-systems are of utmost importance. This model-based approach further facilitates the integration of all other ecosystem aspects to assert the development of sustainable properties, including safety, security, and fairness.

The technical composition of individual constituents in a bottom-up fashion has to be supported by the parallel establishment of a global orchestration, which guides the entire federated ecosystem in the desired direction. This socio-technical challenge demands new and integrated concepts, methods, and tools to define common, multi-modal interfaces of DTs, modularize their building blocks, and ensure their trustworthiness and alignment with business goals.

IV. Illuminating Smart Ecosystem Example Necessitating DT Integration

We use the smart city as an illuminating example to present six use cases (UC) highlighting situations that require the integration of Digital Twins. For each use case, we describe one or more scenarios (identified with black numbered circles ❶–⓬) of how to address the use case and discuss its key corresponding challenges (identified with letters in red circles Ⓐ–Ⓘ). The challenges project a research roadmap to advance support for DT integration. Figure 1, bottom, gives an overview of the mapping of the challenges and scenarios to the aforementioned enabling technologies.

A smart city is one example of a smart ecosystem with a decentralized architecture comprising of several socio-technical systems, such as smart buildings, transportation, farming, but also citizens (Figure 1, top right). To achieve increased levels of service quality, adaptability, and sustainability, these systems will need to interact with each other with unparalleled flexibility and automation. Each system is by itself an ecosystem of smaller parts. To improve modularity and separation of concerns, it is increasingly common to use DTs to realize such ecosystems.

For example, in a controlled environment agriculture indoor farm, a Heating, Ventilation and Air Conditioning (HVAC) DT of a growing room includes a predictive model for room temperature, while a DT of a single plant (e.g., strawberry) may allow simulation of the yield of a fruit.

UC1: Sharing of Building Blocks.
Because the plants influence the temperature of their room, the HVAC's predictive model no longer reflects the physical system. Hence, it is necessary to evolve and adapt the existing HVAC DT by combining it with the Plant DT into a Room DT to perform co-simulation. ❶ **Use of Service.** Assuming that the heat source profile of a plant is known and the HVAC DT exposes a service to estimate room temperature given a known heat source profile, the Room DT could use the service. This means that the HVAC DT was built with reuse in mind and may require the heat source profile to be in a required format. ❷ **Use of Predictive Model.** If the plant's heat source profile is not known, but the HVAC DT exposes its predictive model, then it is possible to ask the HVAC DT to learn an updated predictive model based on the current heat sources in the room (i.e., including the plants). ❸ **Use of Data.** Alternatively, if the HVAC DT allows access to its data (e.g., historical temperature data for a room), then a new predictive model based on the room temperature and the grower's actions could be built for the Room DT to estimate the yield of the room. ❹ **Use of Gateway (Events).** Finally, the new predictive model of the Room DT could be enhanced by feeding it with real-time data by observing the gateway events of the HVAC DT (if exposed).

This use case highlights several integration challenges, namely modularization, standardization of interfaces, and composition under uncertainty.

Ⓐ **Modularization of a DT to enable flexible composition and address challenges related to trustworthiness, economics, and ethics.** A DT is composed of building blocks (gateways, data, models, services) that interact for a given purpose. As a basic design-time challenge relating to many other challenges, it is unclear how the different building blocks need to be modularized to enable flexible composition within/across DTs cost-effectively while addressing trustworthiness.

Across DTs, different DT building blocks may require composition depending on the purpose of the composed DTs. Within a DT, each component may be independently coupled to perform certain actions pertaining to the DT's purpose. A DT can be seen as a composition of grey-box building blocks, each exposed through an interface to interact with the

DT. However, the internal DT modularity is a challenge, as each DT building block may have to be coupled to any other building block seamlessly (Strategies 2a and 2b in Figure 1).

B Well-defined interfaces, model hybridization, data interchange format, and composition operators for DTs and their building blocks. There are no standardized interfaces (APIs) for DTs and their building blocks, which hampers their systematic integration. Hybridization of analytical models (in the classical sense in the model-driven engineering community) for deductive reasoning with statistical and machine learning models (from data sciences) for inductive modeling is a major challenge for leveraging on the combination of such models in DTs. In addition, different DTs and DT technologies imply different representation formats for data, models, and services. Yet for DT integration, the standardization of interchange formats is needed, which may be known for services and data, but currently is difficult for models. For example, the FMI standard [42] facilitates integration by offering the ability to encapsulate models and their simulation, but is limited to the support of analytical models, with no consideration of data and predictive models to combine inductive and deductive reasoning. A recent expert survey by Reif et al. [43] also highlights among other things that standardization and interoperability remain critical challenges. Acharya et al. [44] propose six interoperability levels (technical, syntactic, semantic, pragmatic, dynamic, and organizational) with associated challenges based on a systematic literature review. Furthermore, David et al. [45] report on a panel discussion that stressed the need to reach higher levels of interoperability for DT systems.

C Handling the compounding uncertainty, fidelity, and assumptions that stem from the individual DTs. The trustworthiness of DTs depends on their intrinsic uncertainty, fidelity to their real-world twin, and many other quality factors, all of which must be considered and possibly mitigated using specific countermeasures to ensure the validity of the composed DTs.

UC2: Sharing of DTs.

To support economic expansion, plants other than strawberries can be produced. As such, re-developing an entire DT for the new crops is unreasonable, as rooms, HVACs, and other parts would not change. **5 Plug & Play.** It should be possible to seamlessly replace the Strawberry DT with a Tomato DT. However, if the plug & play capability stays at the technical level (e.g., a standard describing how two DTs can exchange information using a given protocol), then we lose the "semantics" of the Strawberry DT, which is not only a data provider for the room, but also used by growers to better understand the farm. The Digital Twin Consortium also calls for addressing the challenge of plug & play style DTs [46].

D Orchestration or Choreography of DTs. The composition of DTs requires the integration of their building blocks. Depending on the type of composition, integration may only involve the coupling of the DTs' interfaces in a black-box fashion or may require a more complex integration mechanism to couple the DTs as a white-box. For the latter case, this mechanism is still to be defined, as different DTs may require orchestration between models with different timescales, paradigms, communication protocols, data sources, and heterogeneous systems. This mechanism may build on established approaches for service composition.

E Developing specialized techniques to integrate black box, proprietary or legacy DTs. Stakeholder constraints may necessitate the integration of legacy DTs and/or black-box proprietary DTs into the ecosystem. If not developed with integration in mind, using outdated technology and/or supporting multiple versions poses significant integration challenges.

This use case also highlights issues related to **H.a intellectual property** of composed DTs, a general challenge applying to all use cases and discussed at the end of this section.

UC3: Evolution.

DTs may share their building blocks (Strategy 1a in Figure 1), all of which may evolve. For example, in an initial design, each room has its own irrigation system with a tank. The predictive model of the Room DT estimates the level of the solution in the tank by subtracting the quantity already used from the capacity of the tank. In a new design, two instances of a Room DT share the same irrigation tank, which renders the current level estimation of the predictive model invalid. **6 Evolution of Predictive Model.** To correctly estimate the quantity in the tank, the predictive model of the Room DT needs to be evolved. **7 Separation of Concerns.** Alternatively, a new Irrigation Tank DT could be created by extracting it from the Room DT. This new Irrigation Tank DT must then be composed with the two Room DTs. **8 Independent Evolution.** A third alternative is to evolve a Room DT independently of the existing Room DT, i.e., two different variations of the Room DT now exist (one assuming an exclusive tank and one assuming a shared tank).

Similarly, a smart bus company provides open data on bus locations every two minutes. The Bus DT uses open data to access the routes, timetables, etc. and calculates the optimal speed to save fuel. The City DT, on the other hand, predicts emissions based on traffic using open data and cameras installed along streets. When the bus company decides to change the format for its open data and modify its Bus DT, the City DT may be impacted by this change. None of the previous scenarios apply in this case, because it is a technical/internal concern related to the availability of dependent information. Assuming an impact due to the change, a **9 Duplication/Synchronization Mechanism** is required so that each DT has its own access to shared information.

F Support for ad-hoc integration of DTs at run-time. Integrated DTs may require run-time adaptation of their models, services, or gateways, for which there are currently no standards or methods. In particular, the integration and adaptation of "future-time" technologies (i.e., integration of technologies that support what-if scenarios and trade-off reasoning) is an open challenge.

This use case also highlights issues related to **H.b quality assurance and consistency** of composed DTs, which are

general challenges that apply to all use cases and are discussed at the end of this section.

UC4: Opportunistic Integration.
In a smart building, the air conditioning is adapted to the building occupants. To this end, the Building DT has to estimate when people are coming and leaving using some statistical distributions learned from past data. If we connect the Transportation DT to the Building DT, this estimation can be dramatically improved by actual data. This can also happen in smart farms when a new sensor is introduced. For example, the root density of a plant is estimated for what-if analysis purposes, but a chemical sensor can be added to have a more accurate value of the root density. In both cases, we have to adapt the DTs accordingly, and several of the previous scenarios/challenges also apply to this use case.

UC5: Conflict Resolution.
A smart building has a Fire System DT and an HVAC DT. The Fire System DT reads data from the smoke and carbon monoxide sensors, alerts occupants by raising the fire alarm, and calls emergency services. Furthermore, it leads occupants to safety by opening certain doors so that people can leave the building, and closing other doors to avoid fire and smoke propagation. The HVAC DT aims to optimize the energy consumption of the building. On a cold day in the event of a fire, the HVAC DT might want to close the doors to preserve the heat in the building, while the Fire System DT might want to open the doors so that people can leave. In this case, the doors and windows might receive contradictory information. **10 Prioritization.** We need the Fire System DT and HVAC DT to be aware of each other and have a common protocol to communicate and share information so that the HVAC DT does not act on the doors and windows in the event of a fire. We could also have another "piece of software" that deals with the prioritization of DTs.

This use case also highlights issues related to **H.c ontology mismatch and conflicting actuator instructions** in composed DTs – a general challenge discussed at the end of this section.

UC6: Privacy and Ethics.
With the aim of providing fine-grained personalization of services and optimization, a smart ecosystem could equip its citizens with DTs of themselves (or several DTs depending on their roles in the ecosystem). Data and models of such DTs could be hosted locally on the citizen's smartphone, where they keep track of their location and preferences, and might communicate with other smart entities in the environment. For example, the Employee DT of the citizen could communicate that the citizen is claustrophobic to the Building DT so that the Building DT would then assign a room with windows for a planned meeting. **11 Privacy Controls.** On the other hand, this citizen's information should not be revealed to other entities, e.g., employers, managers, or insurance companies, without the citizen's permission. Similarly, an Elevator DT, knowing the passengers' preferences and habits by interacting with their DTs, could display news/commercials/ads that are interesting for the passengers. **12 Ethics Controls.** The composition of the Citizen DTs and the Elevator DT needs to take privacy and ethical considerations into account.

G Ecosystem-aware integration of DTs. Integration of DTs will have to take into account overall ecosystem reality (legal, business, ethics, socio-technical disruptions) during the life cycle of the DTs.

UC6 also highlights issues related to **H.d regulation compliance and privacy** of composed DTs, which are general challenges that apply to many use cases and are discussed next.

H Trust-by-design DTs to ensure an (a) IP-compliant, (b) consistent, safe, secure, (c) non-conflicting, and (d) privacy-compliant ecosystem. Composing DTs requires changing some of their building blocks, which may break or render obsolete existing quality assurance techniques and artifacts. This challenges the reuse and extension of those test suites to ensure the quality of the composed DT. Composing DTs exposes consistency challenges between all building blocks, both vertically (e.g., property preservation between models and services) and horizontally (e.g., contradicting actuation behavior, requiring ontological alignment before integration to ensure interoperability). Furthermore, the composition of DTs may inadvertently expose information, including intellectual property and privacy data. Although the aggregate DTs may comply with regulations and ecosystem policies, their composition may yield contradictory results or may interfere with regulations.

I Develop and deploy the processes, methods, tooling, and competencies to compose the building blocks of DTs for use by multi-disciplinary teams in the ecosystem. Lack of consensus at the ecosystem level on data description, modeling, and service implementation concepts, paradigms, and technologies prevents efficient composition.

V. Conclusion

In summary, using a smart city as an example, we discuss six use cases that highlight scenarios that require the integration of DTs. As a basis for a roadmap for the DT community, we identify nine challenges that need to be addressed to support a systematic approach to DT integration. These challenges relate to A modularization, B interfaces, C uncertainty, D orchestration, E specialized techniques for black box, proprietary, or legacy DTs, F ad-hoc integration at run-time, G ecosystem-aware integration, H trust-by-design, and I processes, methods, tooling, and competencies. We conclude with a call to the DT community to investigate the identified challenges, including the costs, benefits, and risks of DT integration strategies and critical dimensions in multi-stakeholder ecosystems such as privacy, IP, semantic interoperability, and regulatory aspects.

Acknowledgments

This research has been co-funded by the European Union and KDT Joint Undertaking through MATISSE Project GA n. 101140216 and the DFG and ANR project – Model-Based DevOps – 505496753.

References

[1] C. Semeraro, M. Lezoche, H. Panetto, and M. Dassisti, "Digital twin paradigm: A systematic literature review," *Computers in Industry*, vol. 130, p. 103469, 2021.

[2] M. A. Hamzaoui and N. Julien, "Social cyber-physical systems and digital twins networks: A perspective about the future digital twin ecosystems," *IFAC-PapersOnLine*, vol. 55, no. 8, pp. 31–36, 2022.

[3] A. Sharma et al., "Digital twins: State of the art theory and practice, challenges, and open research questions," *Journal of Industrial Information Integration*, vol. 30, p. 100383, 2022.

[4] R. Reine, F. H. Juwono, Z. A. Sim, and W. K. Wong, *Cyber-Physical-Social Systems: An Overview*. Cham: Springer International Publishing, 2021, pp. 25–45. [Online]. Available: https://doi.org/10.1007/978-3-030-76387-9_2

[5] United Nations Sustainable Development Goals, 2025. [Online]. Available: https://sdgs.un.org/goals

[6] A. Singh, A. Kanaujia, V. K. Singh, and R. Vinuesa, "Artificial intelligence for sustainable development goals: Bibliometric patterns and concept evolution trajectories," *Sustainable Development*, vol. 32, no. 1, pp. 724–754, 2024.

[7] B. H. C. Cheng et al., *Software Engineering for Self-Adaptive Systems: A Research Roadmap*. Springer, 2009, pp. 1–26. [Online]. Available: https://doi.org/10.1007/978-3-642-02161-9_1

[8] J. Kephart and D. Chess, "The vision of autonomic computing," *Computer*, vol. 36, no. 1, pp. 41–50, 2003.

[9] R. Eramo et al., "Conceptualizing digital twins," *IEEE Software*, vol. 39, no. 2, pp. 39–46, 2022.

[10] K. Josifovska, E. Yigitbas, and G. Engels, "Reference framework for digital twins within cyber-physical systems," in *2019 IEEE/ACM 5th International Workshop on Software Engineering for Smart Cyber-Physical Systems (SEsCPS)*, 2019, pp. 25–31.

[11] K. Y. H. Lim, P. Zheng, and C.-H. Chen, "A state-of-the-art survey of digital twin: techniques, engineering product lifecycle management and business innovation perspectives," *J. Intell. Manuf.*, vol. 31, no. 6, p. 1313–1337, Aug 2020.

[12] B. Combemale et al., "A Hitchhiker's Guide to Model-Driven Engineering for Data-Centric Systems," *IEEE Software*, vol. 38, no. 4, 2021.

[13] J. Michael, J. Pfeiffer, B. Rumpe, and A. Wortmann, "Integration challenges for digital twin systems-of-systems," in *Proceedings of the 10th IEEE/ACM International Workshop on Software Engineering for Systems-of-Systems and Software Ecosystems*, 2022, pp. 9–12.

[14] T. Olsson and J. Axelsson, "Systems-of-systems and digital twins: A survey and analysis of the current knowledge," in *2023 18th Annual System of Systems Engineering Conference (SoSe)*, 2023, pp. 1–6.

[15] E. Cavalcante, T. Batista, and F. Oquendo, "Exploring synergies and challenges of system-of-systems digital twins," in *2025 IEEE 22nd International Conference on Software Architecture Companion (ICSA-C)*, 2025, pp. 190–196.

[16] A. Bucaioni et al., "Multi-Partner Project: A Model-Driven Engineering Framework for Federated Digital Twins of Industrial Systems (MATISSE)," in *DATE 2025 - Design, Automation and Test in Europe Conference*, Mar. 2025, pp. 1–6. [Online]. Available: https://inria.hal.science/hal-04839759

[17] J. Michael et al., "Integrating models of civil structures in digital twins: State-of-the-Art and challenges," *Journal of Infrastructure Intelligence and Resilience*, vol. 3, no. 3, 2024.

[18] Digital Twin Capabilities Periodic Table, 2025. [Online]. Available: https://www.digitaltwinconsortium.org/initiatives/capabilities-periodic-table

[19] Q. Qi et al., "Enabling technologies and tools for digital twin," *Journal of Manufacturing Systems*, vol. 58, pp. 3–21, 2021.

[20] AWS IoT TwinMaker, 2025. [Online]. Available: https://aws.amazon.com/fr/iot-twinmaker

[21] Azure Digital Twins, 2025. [Online]. Available: https://azure.microsoft.com/en-us/products/digital-twins

[22] M. Neubauer et al., "Architecture for manufacturing-X: Bringing asset administration shell, eclipse dataspace connector and OPC UA together," *Manufacturing Letters*, vol. 37, pp. 1–6, 2023.

[23] B. Combemale, J. Gray, and B. Rumpe, "Model hybridization: towards a unifying theory for inductive and deductive reasoning," *Software and Systems Modeling*, vol. 23, 12 2024.

[24] M. Mernik, J. Heering, and A. M. Sloane, "When and how to develop domain-specific languages," *ACM Computing Surveys (CSUR)*, vol. 37, no. 4, pp. 316–344, 2005.

[25] A. Fuller, Z. Fan, C. Day, and C. Barlow, "Digital twin: Enabling technologies, challenges and open research," *IEEE Access*, vol. 8, pp. 108 952–108 971, 2020.

[26] W. Hu et al., "Digital twin: a state-of-the-art review of its enabling technologies, applications and challenges," *Journal of Intelligent Manufacturing and Special Equipment*, vol. 2, pp. 1–34, 08 2021.

[27] M. Martinelli et al., "Hierarchical digital twin ecosystem for industrial manufacturing scenarios," in *2024 50th Euromicro Conference on Software Eng. and Advanced Applications (SEAA)*. IEEE, 2024, pp. 56–63.

[28] R. Visser, A. Basson, and K. Kruger, "An architecture for the integration of product and production digital twins in the automotive industry," in *ACM/IEEE 27th Intl. Conf. on Model Driven Engineering Languages and Systems*, ser. MODELS Companion '24. ACM, 2024, p. 431–441.

[29] P. Kuruppuarachchi, S. Rea, and A. McGibney, "An architecture for composite digital twin enabling collaborative digital ecosystems," in *2022 IEEE 25th International Conference on Computer Supported Cooperative Work in Design (CSCWD)*, 2022, pp. 980–985.

[30] G. N. Schroeder, C. Steinmetz, R. N. Rodrigues, A. Rettberg, and C. E. Pereira, "Digital twin connectivity topologies," *IFAC-PapersOnLine*, vol. 54, no. 1, pp. 737–742, 2021.

[31] I. Onaji et al., "Digital twin in manufacturing: conceptual framework and case studies," *International Journal of Computer Integrated Manufacturing*, vol. 35, no. 8, pp. 831–858, 2022.

[32] S. Gil et al., "A modeling approach for composed digital twins in cooperative systems," in *2023 IEEE 28th International Conference on Emerging Technologies and Factory Automation (ETFA)*, 2023, pp. 1–8.

[33] S. Gil, E. Kamburjan, P. Talasila, and P. G. Larsen, "An architecture for coupled digital twins with semantic lifting," *Software and Systems Modeling*, 2024. [Online]. Available: https://doi.org/10.1007/s10270-024-01221-d

[34] M. S. Gill, J. Zhang, A. Wortmann, and A. Fay, "Toward automating the composition of digital twins within system-of-systems," in *2024 IEEE 29th International Conference on Emerging Technologies and Factory Automation (ETFA)*, 2024, pp. 1–4.

[35] Y. Fu et al., "Towards the integration of multi-level and multi-view modelling for interoperability," in *2023 ACM/IEEE International Conference on Model Driven Engineering Languages and Systems Companion (MODELS-C)*, 2023, pp. 679–688.

[36] J. Michael et al., "Model-Driven Engineering for Digital Twins: Opportunities and Challenges," *INCOSE Systems Engineering*, 2025. [Online]. Available: https://doi.org/10.1002/sys.21815

[37] M. T. Khedr and J. S. Fitzgerald, "The composition of digital twins for systems-of-systems: a systematic literature review," 2025. [Online]. Available: https://arxiv.org/abs/2506.20435

[38] E. Varela, T. Batista, E. Cavalcante, and A. Almeida, "Pursuing interoperability in digital twins: An analysis of the current research landscape," in *Proceedings of the 40th ACM/SIGAPP Symposium on Applied Computing*. ACM, 2025, p. 803–810.

[39] H. Li et al., *Supporting Digital Twin Integration Using Semantic Modeling and High-Level Architecture*. Cham: Springer International Publishing, 2021, pp. 228–236. [Online]. Available: https://doi.org/10.1007/978-3-030-85910-7_24

[40] M. Pias et al., "On the scaling of digital twins by aggregation," *Data & Policy*, vol. 7, p. e9, 2025.

[41] J. Kienzle, G. Mussbacher, B. Combemale, and J. Deantoni, "A Unifying Framework for Homogeneous Model Composition," *Software and Systems Modeling*, pp. 1–19, Jan. 2019.

[42] FMI standard, 2025. [Online]. Available: https://fmi-standard.org

[43] J. Reif et al., "An expert survey on models and digital twins," 2025. [Online]. Available: https://arxiv.org/abs/2506.17313

[44] S. Acharya, A. A. Khan, and T. Päivärinta, "Interoperability levels and challenges of digital twins in cyber–physical systems," *Journal of Industrial Information Integration*, vol. 42, p. 100714, 2024.

[45] I. David et al., *Interoperability of Digital Twins: Challenges, Success Factors, and Future Research Directions*. Cham: Springer Nature Switzerland, 2025, pp. 27–46. [Online]. Available: https://doi.org/10.1007/978-3-031-75390-9_3

[46] Digital Twin Consort., "Digital twin system interoperability framework," 2021. [Online]. Available: https://www.digitaltwinconsortium.org/pdf/Digital-Twin-System-Interoperability-Framework-12072021.pdf