Digital twins for collaboration and self-integration

Lukas Esterle*, Cláudio Gomes*, Mirgita Frasheri*, Henrik Ejersbo*, Sven Tomforde†, Peter G. Larsen*
Dept. of Electrical and Computer Engineering & DIGIT
Aarhus University, Denmark
{lukas.esterle, claudio.gomes, mirgita.frasheri, hejersbo, pgl}@ece.au.dk
†*Dept. of Computer Science*
Christian-Albrechts-Universität zu Kiel, Germany
st@informatik.uni-kiel.de

***Abstract*—Digital twins are digital representations of cyber-physical systems, allowing bi-directional data and control flow between them. In this paper, we introduce the digital twin concept in self-integrating and self-improving systems. Specifically, we highlight the usage of digital twin models for autonomous systems spontaneously interacting with each other and aiming to improve collaboration and mutual safety. We present the main challenges and an initial architecture to enable autonomous cyber-physical systems to collaborate, self-integrate, and self-improve during run-time.**

***Index Terms*—digital twins, cyber-physical systems, system of systems, autonomy**

I. Introduction

Cyber-physical systems (CPSs), consisting of physical components, interacting with the real world and controlled by a computational system are specifically designed to accomplish a given task [1]. Recently, a novel paradigm has been proposed to control and predict the behaviour of CPS called *digital twins* (DTs) [2], [3]. To realize digital twins of a given CPS an orchestrated co-simulation is often utilised, allowing for simulation of models from different stakeholders [4]. However, enabling multiple CPSs to interact and collaborate with each other still requires corresponding efforts during the design-time and before deployment of the systems. In such a case, each individual system is prepared to collaborate with one another *a priori* by defining how they interact or communicate with each other. While their actions may change, all involved systems are prepared for those changes beforehand or are able to communicate these changes before they occur. While DTs and co-simulation enable us to analyse the collaborative behaviour, we are still required to define all aspects of such studies in advance and establish all parameters before deploying the different systems.

In this position paper, we propose DTs for self-integrating self-improving systems, enabling not only improved performance, but opening up the opportunity to establish collaboration during run-time. The proposed approach is specifically targeted at interwoven systems [5], operating in an open world and opportunistically collaborating where necessary. We further propose an initial framework considering situations where systems are mobile, and not permanently interacting or even in communication and/or range of perception.

Key contributions of this paper are

- Outline two case studies illustrating the main problems and showing the dynamic integration requirements;
- An initial framework to tackle the main problems;
- Challenges faced by autonomous mobile CPS able to interact spontaneously and presenting the underlying benefits of DTs in such situations.

II. Background: Digital Twins

The concept of a DT [6] has emerged as a way to leverage the ever-increasing amounts of data collected from CPSs, fueled by technological advances in sensors, networks, and software. Various definitions of DTs [7]–[10] have been proposed, from when it was first presented by Michael Grieves in 2003 [6].

We view a DT as a system which incorporates different techniques to increase the value of a cyber-physical system, which we denote as Physical Twin (PT). As summarized in Figure 1, if the communication between the PT and the digital system is bi-directional, meaning that data can flow automatically from the physical system to the digital system and vice versa, we call such a digital system the Digital Twin (DT). On the other hand, we call a digital system the Digital Shadow (DS) when the data only flows automatically from the PT to the DT. Naturally, in a DS, data may flow back as well in the broad sense of the word (as in, a human takes action), but this is not done directly.

We can summarize the most common services provided by the DS and DT, detailed and exemplified in [11]:

Visualization: Advances in tools for creating visual interfaces such as Unity (https://unity.com/), Qt (https://www.qt.io/), Grafana, Dash, Gazebo, and so on, have made it easier to create intuitive interfaces. More details about data visualization can be found in [12];

State Estimation: The state represents the internal condition or status of a system at a given instant of time. It is seldom possible to employ sensors to measure all states of the system. State estimation is the use of different methods to estimate the states of interest that cannot be obtained directly through measured data. More details can be found in [13].

Monitoring: Monitoring in the context of a DS is the act of observing and evaluating the behaviors of a PT as it

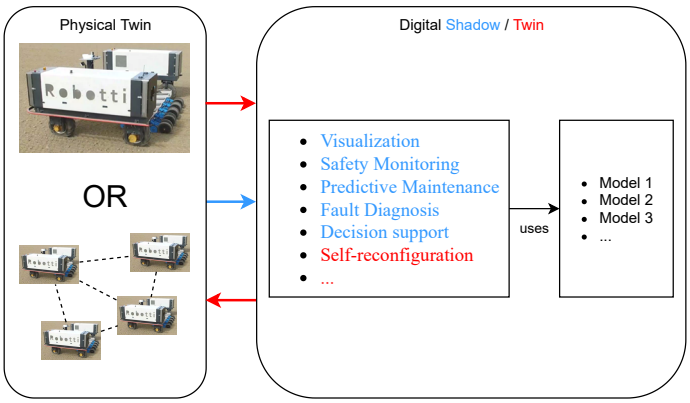

Fig. 1. Comparison of Digital Twins and Digital Shadows and their relations to the Physical Twin, here depicted using the agricultural robot example. A Digital Shadow represents information from the Physical Twin (blue) while the digital twin can, in addition, also act upon the Physical Twin (red).

operates [14]. Monitors frequently include state estimators, and form the backbone of anomaly detection [15]. We refer the reader to the state of the art in run-time monitoring [16], [17].

What-if (co-)simulation: What-if simulation is a data-intensive simulation whose goal is to inspect the behavior of the PT under some given hypotheses called scenarios [18]. A what-if simulation could try alternative interventions purely in a virtual setting to inspect what the consequences would be, *before* taking a final decision about what intervention would be best. In cases where the simulations consist of multiple sub-models, a co-simulation could be used [4], [19].

Self-Adaptation: Self-adaptation is the ability of a computer system to change parts of all of its working algorithms over time [20]–[22]. Self-adaptation procedures can be very intricate, rely on the other DT services, rely heavily on domain knowledge, and can be application-specific.

Most of the above services rely on models of the PT, as illustrated in Figure 1. An obvious example is the what-if (co-)simulation. Moreover, state estimation, when implemented with, e.g., a Kalman Filter, also relies on models. Figure 1 also identifies other services such as fault diagnostics, decision support, safety monitoring, or predictive mainentance (to mention a few) that might be of relevance when using DTs. However, the 5 services mentioned above are most common across DT research [11].

Most models, if simulated independently from the measured system behavior, even if initialized to the exact same initial state as the PT, will naturally drift apart, due to noises in sensors and actuators, and other physical phenomena that are not captured by the model (exceptions to this rule are periodic systems for which the simulation may too be periodic) [23]. As a consequence, a simulation is as good as how recently it was re-initialized to the measured state of the system.

This is why every DT must include mechanisms to detect when a model has drifted too much and re-initialize it to a correct state, or, in the case that the PT changed so much that the model is no longer valid, to decommission the model (or re-calibrate its parameters, if that is possible). We shall denote such a component, as the "Model Manager".

The "Model Manager" can also be responsible for selecting the right model, for each requested service. This is because models have varying levels of fidelity and goals. For example, a model used in path planning of robots operating in an agricultural field does not need to include every equation relating to inertial forces in the system. Instead, the main geometry and kinematics of the robot will suffice.

Furthermore, DTs can be generated and operated at different scales. Consider having a DT of the engine of a robot, a robot itself may have multiple engines and this combination of multiple DTs is represented as another DT altogether. Multiple robots can be operating in a collective or as a system-of-systems, creating another DT out of multiple DTs. Again, this aggregation and generalisation shall be handled by the "Model Manager".

III. Scenarios

The "Self-Improving Systems Integration" (SISSY) initiative [24] focuses on the question of how individual subsystems can autonomously decide about their integration into an overall system constellation and which basic infrastructure has to be provided for the efficient operation of the resulting complex structures [25]. In other words, SISSY proves techniques and solutions to automatically control and manage system-of-system constellations [26] that are composed of a variety of heterogeneous distributed elements – typically referred to as so-called Interwoven Systems [5].

In the following, we describe two SISSY scenarios consisting of autonomous systems self-integrating in their environment and aiming to improve their performance over time. When two or more systems encounter each other, they can engage in mutual interactions. Here we distinguish between direct and indirect interactions. Direct interactions encompass activities that require active exchange of information, tasks, or physical resources. Indirect interactions, on the other hand, can occur when CPSs occupy the same physical space. In both cases, the encounter can affect the behaviour of the

systems involved further affecting safety and performance of each individual CPS.

In the following, we use the set $S = \{s_1, s_2, ...s_n\}$ to denote a set of systems, and we distinguish between systems that are DT-enabled, denoted by $C \subseteq S$, from systems that are not DT enabled $P \subset S$. Note that any system in S can be a CPS, however, not all CPS are automatically DT enabled, as DT enabling a system presents unique challenges regarding reconfigurability, security, and safety [11].

Each DT-enabled system $s_i \in C$ has the corresponding DT, denoted as $d_i \in D$, where D is the set of digital twins. As we show in the scenarios presented next, DTs can support the interaction of systems to perform more efficient, safe, and accurate. We distinguish two possible interactions:

Model Exchange – Happens when a DT d_i requests d_j (where $d_i, d_j \in D$) to provide the former with a model. The DT d_i can then use the model obtained to simulate the behavior of s_j.

Service Request – Happens when a DT d_i requests d_j (where $d_i, d_j \in D$) to execute some service, such as performing a specific action, requesting specific data, or running a simulation of s_j.

Both these interactions can attain the same goal. However, they differ in the challenges and complexity in implementing the DTs.

A. Farming robots

The first scenario comprises a set of agricultural robots, operating in a specific field. In this first scenario, we consider a set of homogeneous robots, where all robots are able to complete the same set of tasks and can communicate with each other when within WiFi range of one another. Furthermore, robots belong to the same stakeholders, meaning, all robots can trust each other when exchanging data, and goals are known and shared among all individuals. More so, robots from the same manufacturer can exchange models without worrying about intellectual property problems.

Due to the homogeneity, all robots may have the DT models of all other robots in the environment *a priori* or receive them during runtime. In both cases, they are aware of the underlying semantics of each model (i.e., what is this model representing and for what is it used). In addition to the set C of robots, this scenario might also have systems P involved. This can be humans, animals, and other machines such as tractors or combined harvesters. Importantly, these physical systems are not DT-enabled, meaning they do not possess a DT or the ability to share information by digital means with a CPS.

While this first example builds upon these parameters defined earlier, it is still rather simple as robots have *a priori* knowledge and are able to communicate directly with each other. The latter could become a necessity when they are planning the routes they are taking and trying to avoid collisions with other systems. This is easier, or at least feasible, with CPSs, but much harder with systems lacking an interface for direct information exchange. Robots can furthermore update the models they possess of others either by replacing them on each (physical) encounter or by incorporating historical data. When robots are deployed on the field, they have a common task such as plowing, reaping, or seeding a field. The tasks and routes for all robots can be predefined initially. However, when new robots enter the field, the tasks and routes do not need to be recalculated and re-assigned but can be adapted based on encounters during execution, alleviating the need for a central controller. In a similar way, if a robot fails to complete a task, another robot can take over by receiving the relevant information for task completion from the other system.

B. Ferries

The second scenario considers autonomous ferries, $s_i \in C$, operating in a shared environment, but controlled and handled by different stakeholders. In contrast to the first scenario, model exchange may be difficult due to intellectual property constraints. However, service requests, such as change in velocity, providing planned travel paths, or performing speed adjustments, are a valid alternative to model exchanges. Specifically, when they enter tight bodies of water, collaboration can be required among the different ships and their respective DTs. Having a DT of the other system can support this effort. However, since ships are often operated by different stakeholders, they might not be able to communicate and exchange models directly. This leaves a CPS to generate a DT based on observations alone, besides simple service requests. In addition to other autonomous ferries $s_i \in C$, our system may again encounter other obstacles such as other boats, ships, or water sportsmen where $s_j \in P$. This requires each system to develop the DT model of other systems in the environment through observation, introducing an additional challenge. With each following encounter, the autonomous ferries can retrieve the previously developed modes for calibration and refinement. If the different systems are able to communicate with each other, they can exchange historical data as in the previous scenarios with robots for arable tasks. If they are not, either because the system $s_j \in P$ or because they do not have common means of communication, they have to rely again on sensed data for calibration.

C. Other scenarios

One can also conceive alternative scenarios for example in the industrial or the smart grid domain. The future industry is expected to manufacture and produce highly personalised products requiring the interaction and collaboration of a broad spectrum of types of machinery (cp. agile tooling and additive manufacturing). Additionally, the composition of robotic components from different stakeholders for specialised production might be required. In smart grid scenarios, a variety of systems have to adjust their operations to balance their energy consumption across the entire grid. These machines can be held by a single or multiple stakeholders. For such an autonomous combination of components and machines to achieve a common goal, DTs can play a key role in identifying and selecting the optimal setting.

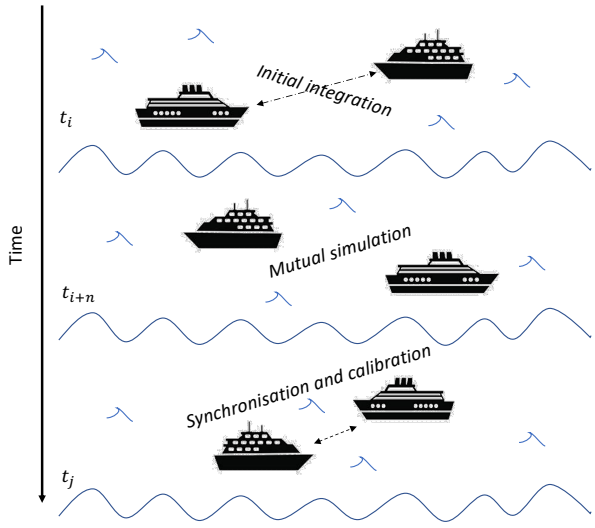

Fig. 2. Systems A and B meet. They exchange plans and digital models, if available, if not, they create one based on observations. While in range, they calibrate their models. At time t_i they are out of range and perform mutual simulations about their behaviour. When they meet again at time t_j they resynchronise their models, identify errors and recalibrate.

IV. Challenges

With the opportunities brought about by DTs for SISSY systems, several challenges need to be tackled to fully capitalise on the accompanying benefits. We categorise the challenges into "Inherent challenges in Digital Twins", affecting any system utilising a digital twin and "Challenges of interacting systems using DTs", specifically considering interacting and collaborating systems such as SISSY systems.

A. Inherent challenges in Digital Twins

Calibration of DT models is necessary when initial models are not representing the real physical system accurately enough. The required data for calibration can be acquired through observation or information exchange. Verification of the received data becomes an additional challenge in the process. Self-integrating and improving systems need to calibrate their models in order to operate on correct assumptions but also to incorporate the changes inflicted by others.

Reliability of different models needs to be estimated during operation, especially for those models possibly fitting to the observations of others. This can further be turned into an ensemble learning approach, where various models are considered jointly to decrease the inherent uncertainty for predicted behaviour. Without such reliability, integration processes are subject to unnecessary uncertainty.

Self-improvement using Machine Learning is used for optimising the selection of models over time. This then serves as mechanism to decrease uncertainties and allow for a context-dependent model choice. Systems need to differentiate between learning about themselves and learning about their environment in order to optimise selection processes.

Autonomous generation of DT models is a key challenge for handling large-scale, context-dependent, and multi-entity scenarios as especially given in the SISSY context. The challenge comprises various aspects ranging from customisation of a prototype model to generative modelling from observations and to transfer learning for DT technology.

Semantic reasoning about models is required when models are utilised for specific purposes. For example, receiving a kinematic model will not be suitable to establish the energy consumption of the respective physical twin. This requires a deeper understanding and a causal connection between models and the concomitant physical twins.

Trust in other models when systems exchange information, a question of trust between machines arises [27]. This requires the system to verify the information provided during run-time and the short time frame of interaction.

B. Challenges in interacting systems for DT

Comparison of two models can become relevant when a system encounters another system and needs to establish whether a DT model is already available for this type of system. This allows not only to ensure we do not duplicate a model for a specific system but also to generate and build upon generic base models to generate models for a newly encountered system. This challenge is typically relevant when receiving a model from another system.

Dynamic integration of models also known as *hot swapping* is required when new DTs become available during runtime. The systems need to replace currently utilised DTs with the new DT and incorporate all previously available information. In addition, a smooth transition or hand-over between DTs needs to be guaranteed. The challenge arises when receiving a model from another system.

Run-time maintenance of models maintaining a large number of models not only requires respective resources for storing individual models but also coordination on how to keep them up-to-date. This becomes more challenging when multiple systems are in the vicinity, requiring updates in multiple models concurrently. Model exchange specifically raises this challenge.

Selecting a specific model from a collection of models based on observations of physical counterparts (incl. determination of triggers to change the model at runtime, e.g. based on anomaly detection). This occurs when direct model exchange is not possible and service requests are required. Local models can support selecting and requesting adequate services.

Service representation of a DT is required in a way that can be interpreted by other systems. This will enable and facilitate interaction and collaboration. Specifically when other systems require specific services. Furthermore, when models are exchange, this ensures other systems know what services can be requested or expected.

Verification of actions or services performed by other systems during runtime to ensure safe and efficient interactions. This requires verification at runtime based on expected outcomes and predefined rules and is relevant in model exchange and service request situations. However, verification can be specifically hard when the verifying system can only rely on observations during service requests [28].

Housekeeping of models is not only required but also needs to be done intelligently for two main reasons. First, mobile cyber-physical systems might only have a limited amount of available resources to store, process, and retrieve digital models. This requires the system to ensure to only keep those models really required. Second, a system cannot know which DT models it will require in the future. Approaches are required allowing to retain a diversity of DT models where possible but also keep those required for frequent interaction updated and well organised.

V. Architecture

While DTs require a wide variety of functionalities, we envision DTs for SISSY systems to incorporate only a few, very specific components. In detail, a SISSY system to fully capitalise on the benefits of DTs is required to have the following components (cp. Figure 3).

- **Interface:** The *interface* comprises sensors and actuators which the system can use to interface with the real world. Sensors allow the system to perceive its environments including other systems, their behaviour, and actions. Actors, on the other hand, allow the system to manipulate and affect the environment. As with autonomous systems, the actions performed by the system are based on plans developed by the planner. Finally, the *interface* allows for communication and data exchange. This includes the exchange of current and historical data but also the exchange of the individual systems' digital twin model.
- **DT DB:** The *Digital Twin Database* contains all known DTs. The system can store and recover DT models in this database.
- **Translator & Comparator:** The *translator* enables the DT Manager to translate incoming models and assign the different semantics to corresponding signals. The *comparator* is able to compare different DT models. This allows the system to check whether specific DT models are already available in the database and which ones need to be created from scratch. While the *comparator* is initially conceived as an approach to compare models directly, one can also imagine the *comparator* to infer comparisons based on sensory data rather than direct model comparison. The *comparator* will return a distance metric between two models or a model and perceived information.
- **Creator & Calibrator:** The *creator* generates new DT models. This can be performed from scratch or using a previously perceived model as a bootstrap. The *calibrator* calibrates inaccurate models, diverging from the real world in order to be realigned and reflect the real world again. The *calibrator* is on one hand able to identify such divergence but also to rectify them. Calibration can be performed based on current observations as well as based on historical data received from the other system.
- **Planner:** The *planner* utilises DT models to calculate ongoing as well as future interactions with another system. While using information from and acting on the world through the interface, the planner also utilises the *digital twin manager* comprised of the *calibrator*, the *comparator*, the *creator* and the *Digital Twin Database*.
- **Destroyer:** as DT models become outdated, the system is not required to keep the data anymore. The *destroyer* component will make decisions about the time of destruction of DT models in order to keep the amount of available DTs limited. Removing DT models requires reasoning about the potential requirement of the respective model in the future. This can be for direct interactions or utilising the DT model as a blueprint when encountering new systems to interact.

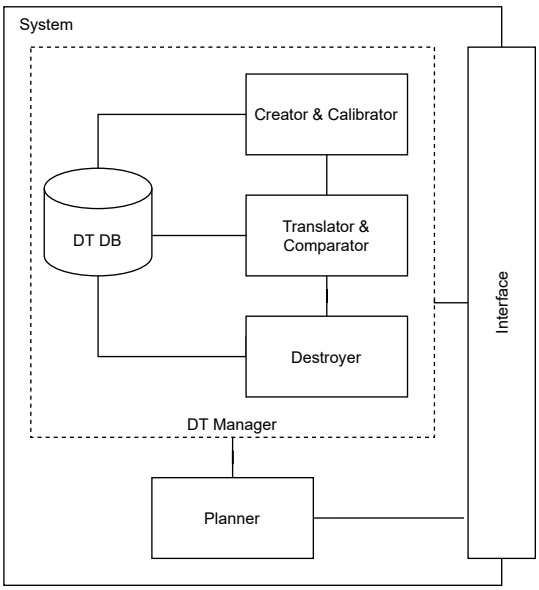

Fig. 3. Potential architecture of a CPS utilising DTs and DT integration at runtime. When sensing another system, the *Translator & Comparator* tries to find a matching model in its database. If not available, the comparator compares against its own DT. If this is still no match, the *Creator & Calibrator* is triggered, building a new DT based on the closest available DT. One might consider a baseline DT with minimum information to start from. Models currently in use are refined by the *Calibrator* before feeding it back into the database. All available models are given to the *Planner* to make decisions on the interactions potentially considering performance and safety constraints.

VI. Discussion

Current work on utilising DTs mainly focuses on a single-system-perspective, i.e. an optimised DT-based control strategy of an individual CPS. In this paper, we proposed to enlarge the focus towards making DT a key enabler for SISSY. Specifically as a tool to support autonomous and self-

motivated integration of individual systems into an overall system constellation. Based on a discussion of the aspects of DTs, we introduced two basic scenarios that emphasise how DT technology could be beneficial in SISSY applications.

As a basis for appropriate utilisation of DT technology, we presented an architectural concept and explained the fundamental concepts. This led to an attempt of deriving challenges to be tackled towards DT-based self-integration, ranging from an initial automated generation of DT to run-time maintenance of models and to trusted exchange of models. In order to fill the architectural components, we start with an investigation of how to generate, store, and adapt DT models at run-time. This then serves as a basis for optimising the self-integration decision of systems such as the farming robots or the autonomous ferries introduced in the presented scenarios.

ACKNOWLEDGMENT

This research has been partly funded by the following research grants:

- "AgroRobottiFleet" project by Innovationsfonden Denmark.
- ITEA 3 and Innovationsfonden Denmark in "Unleash Potentials in SIMulations (UPSIM)" (19006).
- The Poul Due Jensen Foundation with the Digital Twins for Cyber-Physical Systems (DiT4CPS) project.
- German Ministry for Transport and Digital Infrastructure within the project "CAPTN FördeAreal - Erprobung einer (teil-)autonomen, emmissionsfreien Fährschifffahrt im digitalen Testfeld" (45DTWV007B).

REFERENCES

[1] L. Esterle and R. Grosu, "Cyber-physical systems: challenge of the 21st century," *e & i Elektrotechnik und Informationstechnik*, vol. 133, no. 7, pp. 299–303, 2016.

[2] F. Tao and Q. Qi, "Make more digital twins," 2019.

[3] J. S. Fitzgerald, P. G. Larsen, and K. G. Pierce, "Multi-modelling and co-simulation in the engineering of cyber-physical systems: Towards the digital twin," in *From Software Engineering to Formal Methods and Tools, and Back - Essays Dedicated to Stefania Gnesi on the Occasion of Her 65th Birthday*, ser. Lecture Notes in Computer Science, vol. 11865. Springer, 2019, pp. 40–55.

[4] C. Gomes, C. Thule, D. Broman, P. G. Larsen, and H. Vangheluwe, "Co-simulation: a survey," *ACM Computing Surveys (CSUR)*, vol. 51, no. 3, pp. 1–33, 2018.

[5] K. Bellman, S. Tomforde, and R. P. Würtz, "Interwoven systems: Self-improving systems integration," in *Proceedings of the International Conference on Self-Adaptive and Self-Organizing Systems Workshops*. USA: IEEE Computer Society, 2014, pp. 123–127.

[6] M. Grieves and J. Vickers, "Digital Twin: Mitigating Unpredictable, Undesirable Emergent Behavior in Complex Systems," in *Transdisciplinary Perspectives on Complex Systems: New Findings and Approaches*. Cham: Springer, 2017, pp. 85–113.

[7] M. Liu, S. Fang, H. Dong, and C. Xu, "Review of digital twin about concepts, technologies, and industrial applications," *Journal of Manufacturing Systems*, vol. 58, pp. 346–361, 2021.

[8] A. Rasheed, O. San, and T. Kvamsdal, "Digital Twin: Values, Challenges and Enablers From a Modeling Perspective," *IEEE Access*, vol. 8, pp. 21 980–22 012, 2020.

[9] F. Tao, H. Zhang, A. Liu, and A. Y. C. Nee, "Digital Twin in Industry: State-of-the-Art," *IEEE Transactions on Industrial Informatics*, vol. 15, no. 4, pp. 2405–2415, 2019.

[10] F. Tao, Q. Qi, L. Wang, and A. Y. C. 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.

[11] H. Feng, C. Gomes, C. Thule, K. Lausdahl, A. Iosifidis, and P. G. Larsen, "Introduction to Digital Twin Engineering," in *The Annual Modeling and Simulation Conference*, Virginia, USA, 2021, p. to appear.

[12] W. W.-Y. Chan, "A survey on multivariate data visualization," *Department of Computer Science and Engineering. Hong Kong University of Science and Technology*, vol. 8, no. 6, pp. 1–29, 2006.

[13] D. Simon, *Optimal State Estimation: Kalman, H Infinity, and Nonlinear Approaches*. John Wiley & Sons, 2006.

[14] E. Bartocci, J. Deshmukh, A. Donzé, G. Fainekos, O. Maler, D. Ničković, and S. Sankaranarayanan, "Specification-Based Monitoring of Cyber-Physical Systems: A Survey on Theory, Tools and Applications," in *Lectures on Runtime Verification*. Cham: Springer, 2018, vol. 10457, pp. 135–175.

[15] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey," *ACM Computing Surveys*, vol. 41, no. 3, pp. 15:1–15:58, 2009.

[16] Yilin Mo, T. H.-J. Kim, K. Brancik, D. Dickinson, Heejo Lee, A. Perrig, and B. Sinopoli, "Cyber–Physical Security of a Smart Grid Infrastructure," *Proceedings of the IEEE*, vol. 100, no. 1, pp. 195–209, 2012.

[17] K. Paridari, N. O'Mahony, A. El-Din Mady, R. Chabukswar, M. Boubekeur, and H. Sandberg, "A Framework for Attack-Resilient Industrial Control Systems: Attack Detection and Controller Reconfiguration," *Proceedings of the IEEE*, vol. 106, no. 1, pp. 113–128, 2018.

[18] S. Rizzi, "What-if analysis," in *Encyclopedia of Database Systems*. Springer, 2009, pp. 3525–3529.

[19] R. Kübler and W. Schiehlen, *Multibody System Dynamics*, vol. 4, no. 2/3, pp. 107–127, 2000.

[20] A. Sari and J. C. Sopuru, "Bayesian Model for Evaluating Real-World Adaptation Progress of a Cyber-Physical System," in *Artificial Intelligence Paradigms for Smart Cyber-Physical Systems*. IGI Global, 2021, pp. 324–343.

[21] P. Zhou, D. Zuo, K. M. Hou, Z. Zhang, J. Dong, J. Li, and H. Zhou, "A Comprehensive Technological Survey on the Dependable Self-Management CPS: From Self-Adaptive Architecture to Self-Management Strategies," *Sensors*, vol. 19, no. 5, p. 1033, 2019.

[22] D. Weyns, "Software engineering of self-adaptive systems," in *Handbook of Software Engineering*. Springer, 2019, pp. 399–443.

[23] G. Lumer-Klabbers, J. O. Hausted, J. L. Kvistgaard, H. D. Macedo, M. Frasheri, and P. G. Larsen, "Towards a digital twin framework for autonomous robots," in *SESS: The 5th IEEE Int. Workshop on Software Eng. for Smart Systems*, COMPSAC 2021. IEEE, July 2021.

[24] K. L. Bellman, C. Gruhl, C. Landauer, and S. Tomforde, "Self-improving system integration - on a definition and characteristics of the challenge," in *IEEE 4th International Workshops on Foundations and Applications of Self* Systems, FAS*W@SASO/ICCAC 2019, Umea, Sweden, June 16-20, 2019*, 2019, pp. 1–3.

[25] K. L. Bellman, J. Botev, A. Diaconescu, L. Esterle, C. Gruhl, C. Landauer, P. R. Lewis, A. Stein, S. Tomforde, and R. P. Würtz, "Self-improving system integration - status and challenges after five years of SISSY," in *2018 IEEE 3rd International Workshops on Foundations and Applications of Self* Systems*, 2018, pp. 160–167.

[26] M. W. Maier, "Architecting principles for systems-of-systems," *Systems Engineering: The Journal of the International Council on Systems Engineering*, vol. 1, no. 4, pp. 267–284, 1998.

[27] P. Andras, L. Esterle, M. Guckert, T. A. Han, P. R. Lewis, K. Milanovic, T. Payne, C. Perret, J. Pitt, S. T. Powers *et al.*, "Trusting intelligent machines: Deepening trust within socio-technical systems," *IEEE Technology and Society Magazine*, vol. 37, no. 4, pp. 76–83, 2018.

[28] L. Esterle, B. Porter, and J. Woodcock, "Verification and uncertainties in self-integrating system," in *2021 IEEE International Conference on Autonomic Computing and Self-Organizing Systems Companion (ACSOS-C)*, 2021.