EA Blueprint: An Architectural Pattern for Resilient Digital Twin of the Organization

Chapter · September 2021

DOI: 10.1007/978-3-030-86507-8_12

Farid Edrisi
Linnaeus University

Diego Perez-Palacin
University of Zaragoza

Mauro Caporuscio
Linnaeus University

EA Blueprint: An Architectural Pattern for Resilient Digital Twin of the Organization

Farid Edrisi[1(✉)], Diego Perez-Palacin[1], Mauro Caporuscio[1], Margrethe Hallberg[2], Anton Johannesson[3], Claudia Kopf[2], and Johanna Sigvardsson[3]

[1] Linnaeus University, Växjö, Sweden
{farid.edrisi,diego.perez,mauro.caporuscio}@lnu.se
[2] Scania AB, Oskarshamn, Sweden
{grethe.hallberg,claudia.kopf}@scania.se
[3] Virtual Manufacturing AB, Göteborg, Sweden
{anton.johannesson,johanna.sigvardsson}@virtual.se

Abstract. Advancements in Cyber-Physical Systems, IoT, Data-driven methods, and networking prepare the adequate infrastructure for constructing new organizations, where everything is connected and interact with each other. A Digital Twin of the Organization (DTO) exploits these infrastructures to provide an accurate digital representation of the organization. Beyond the usual representation of devices, machines and physical assets supplied by Digital Twins, a DTO also include processes, services, people, roles, and all other relevant elements for the operation of organizations. Therefore, DTO can play a key role in realizing and analyzing aspects of organizations, assisting managers on the knowledge of the organization status, and foreseeing possible effects of potential changes in the organization. However, due to the dynamic, open, and ever-changing environment of organizations, an established DTO will soon degrade or even lose all its utility. Therefore, a DTO needs to evolve to recover its utility when the organization changes. The development of flexible, resilient, and easy to evolve DTO has not been well-addressed yet. Most of the existing proposals are context-dependent, system-specific, or focus on providing solutions in high-level abstraction. This work leverages Enterprise Architecture to propose an architectural pattern to serve as a blueprint toward the development of resilient DTO.

Keywords: Resilient Digital Twin of Organization · Enterprise architecture · Architectural pattern

1 Introduction

Organization's structures and processes are affected by changes in the environment, e.g., the market or regulations. Organizations need to adapt quickly

R. Adler et al. (Eds.): EDCC 2021 Workshops, CCIS 1462, pp. 120–131, 2021.
https://doi.org/10.1007/978-3-030-86507-8_12

to these forces, in order to be resilient and mitigate risks while taking advantage of opportunities they create: *forces drive the velocity of change.* A resilient organization should be therefore agile and responsive to changes, continuously adapting its capabilities, processes, strategies and information to meet changing objectives.

In the digitalization era, a Digital Twin of the Organization (DTO) plays a key role, as it allows for continuously simulating, analyzing, and monitoring the organization. DTO is defined as an accurate digital representation of the organization [20]. That is, DTO represents all the organizational elements (e.g., devices, processes, services, and people) as models that can be continuously simulated and analyzed in continuous assessment and optimization of the organization [14].

In order to fully benefit from DTO, the physical organization and developed DTO must be aligned. However, the continuous change of organizations creates misalignments between the physical organization and DTO, which reduces the DTO utility and acceptance within the organization.

In an ideal situation, evolutions in the physical organization and DTO would occur simultaneously and misalignments would not exist. However, in real situations, misalignments frequently happen. When the forces drive the change to actuate, neither physical organization can wait for the evolution of the DTO to execute its change, nor the evolution needed in the physical organization is completely well-defined in advance for the DTO developers. This creates the necessity of building *resilient DTO* which can cope with the disturbance in the alignment between DTO and physical organization caused by the changes in the organization.

Although the literature presents extensive discussions about the possible applications of DTO, there is still a lack of established engineering practices for architecting, developing, and operating flexible, resilient, and easy to evolve DTO. This prevents industry to fully benefit from DTO, as available solutions are either context-dependent or system-specific and challenging to adapt, maintain, and evolve.

There exist major open challenges in the construction of resilient DTO. We have previously discussed a set of architectural issues when developing a DTO [5], and have argued the need of defining an Architectural Framework that includes: (1) an *Architectural Pattern* achieving flexibility, (2) a *Reference Model* decomposing functionality into architectural elements, and (3) a *Reference Architecture* mapping the Reference Model onto physical elements. This work aims to address the first point by proposing an *Architectural Pattern* to facilitate the development of resilient DTO. According to the resilience classification in [3,4], it focuses on DTO resilience as *flexibility* (this property is also called flexibility in [16], and it is similar to the graceful extensibility and sustained adaptability properties in [21]).

To this end, we leverage *Enterprise Architecture* [17] (EA) to serve as a blueprint for developing the resilient DTO. In fact, EA is by default positioned to play a key role in realizing a DTO, as it embeds all the *principles* and *models* used in the realization of an organization: the *Business architecture* provides the structural and behavioral models, whereas the *Information Architecture* provides the data representing the actual status of the organization. The proposed architectural pattern, named EA Blueprint Pattern, fosters the use of (*i*) Service-Oriented Architecture (SOA) for instantiating the *Business Architecture*, and (*ii*) Model-View-Controller (MVC) for instantiating the *Information Architecture*. Further, the architectural pattern includes an additional element serving as a synchronization and alignment point in between digital models and real-time data coming from the real world.

The remainder of this paper has been organized as follows. Section 2 introduces the context of the resilient DTO and an example. Section 3 presents the proposed architectural pattern. Section 4 reports recent related works regarding Digital Twin architecture. Finally, Sect. 5 concludes the paper by addressing future research directions.

2 Resilient DTO Context

This works adopts the concept of system resilience defined in [7], which builds upon the concept of dependability, *the persistence of system dependability when facing changes*. We refer to types of changes in the organization using the terminology in the resilience framework in [4], which also conforms to the definition in [7].

The resilience framework in [4] distinguishes two kinds of changes that affect systems resilience, and two additional dimensions for the changes that are orthogonal to the two kinds of changes.

The two kinds of changes are called *operational changes* and *evolutionary changes*. Operational changes are changes that cause a modification of the system state. That is, the required DTO functionality remains the same, but its operational conditions are different. Evolutionary changes are changes that cause a modification of the acceptance criterion of the DTO; i.e., the DTO state can remain the same, but its expected functionality has evolved. The architectural pattern proposed in this work focuses on the DTO resilience when facing evolutionary changes.

The two orthogonal dimensions are the *readiness* of the system to cope with a change, and the *persistence* of the change in the system. Regarding the *readiness*, this work helps to deal with evolutionary changes that are *unexpected* or *expected-unhandled*; i.e., the current DTO is not equipped with mechanisms to handle them. Expected unhandled changes can happen when an organization explores a new business opportunity. Even if that is a planned event and the change is expected, there is high uncertainty how the organization will new develop and consolidate the new business branch and, thus, the DTO cannot be ready and aligned to the physical organization when the evolutionary change happens. In turn, examples of unexpected evolutionary changes in the organizations can be: forced quick changes in the organization business to endure new market trends and avoid going out of business, or the change in functionality of a public hospital when it must handle a surprising pandemic. Regarding the *persistence* of the change, this work focuses on *permanent* changes; i.e., the DTO will need to evolve because the evolutionary change will not disappear in a reasonable time horizon.

Since not all evolutionary changes cause the same level of damage or misalignment in the DTO, different DTO evolution tasks should be able to get different priority. An evolutionary change that causes a "minor" damage can still allow the DTO to provide some acceptable functionality even if not fully aligning with the physical organization. However, an evolutionary change in the physical organization can also cause a "major" damage in the DTO (e.g., restructuring the behavior or processes in the organization, adding new functionalities or business branches, or changing the format of input data streams). In that case, the DTO will soon degrade or even lose all its utility and functionality. Therefore, the DTO needs to evolve with high priority to regain its utility in the organization. Since several evolutionary changes can simultaneously coexist in the organization, it is necessary to build DTO whose parts can evolve independently. This allows prioritizing the evolution of the parts of the DTO that caused the greatest degradation in the DTO functionality.

It is worth noting that this work concentrates only on DTO that are resilient to evolutionary changes in the physical organization. The utilization of DTO for improving the organization's resilience is another very interesting and challenging research topic which it is out of the scope of this work.

2.1 Example of Resilient DTO

The utilization of Digital Twins has already been extensively reported for cases of devices, machinery and manufacturing processes in production plants. The proposed architectural pattern for DTO partially learns from experience in those domains, but the utilization of DTO is not restricted to organizations that are manufacturing plants or industry. To illustrate the DTO resilience challenges that exist in other types of organizations, the following paragraphs show an example where the organization is a public hospital.

The hospital consists of various departments or sections offering acute services, e.g. an emergency department, burn unit, surgery, and urgent care. It also includes more specialist units like cardiology or coronary care unit, intensive care unit, neurology, and obstetrics and gynecology, etc. Moreover, there are several roles of staff working in the hospital, such as doctors, nurses, janitors, administrators, etc. A DTO is already in place and represents all the hospital processes and elements aforementioned. It helps to dynamically optimize the hospital operations, scheduling, and utilization of resources. The DTO is also used by management to estimate the free resources in the near future in order to notify nearby hospitals about its capabilities to accept their patients in the next days, if necessary.

In 2020, the Covid-19 pandemic strikes hard. The hospital has to evolve its functionality quickly. Dedicating more staff, wings, beds and other types of equipment to Covid-19 patients, changing work shifts, canceling non-emergency surgeries, separating intensive care unit into sections to minimize infections, etc.

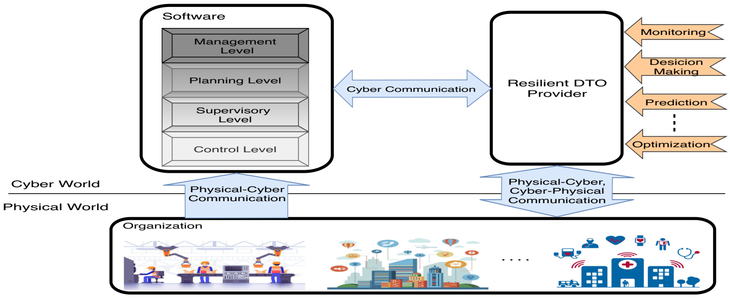

Fig. 1. DTO high-level view

are some of the required changes. In addition, tracking the patients' pathway and monitoring their situations frequently for taking necessary actions in major risks are other new tasks for the hospital managers. In this pandemic situation, a DTO would offer very useful functionality and support to the hospital new operations.

However, since there has been an unexpected evolutionary change in the organization, the current DTO is misaligned and, thus, it is almost useless in assisting the new activities that are critical for the hospital. The DTO must evolve. A resilient DTO would be able to evolve quickly.

3 The EA Blueprint Pattern

To help the development of resilient DTO, this section presents the EA Blueprint Pattern, an architectural pattern aiming at facilitating resilient DTO development and evolution.

Context: You are developing the resilient DTO Provider (see Fig. 1) mirroring the structure and behavior of a given organization by modeling and receiving the data of its physical and cyber elements to use in various applications. Since the organization is in continuous change and evolution, frequent misalignment could happen between DTO and real organization. So, DTO must be flexible and able to change and evolve accordingly.

Problem: How to develop a resilient DTO to evolve it according to the organization needs?

Forces:

- Resilient DTO elements should be developed according to Agile Design Principles [10] – i.e., Single responsibility, Open/closed, Liskov substitution, Dependency inversion, and Interface segregation –, so that they can be developed, deployed and evolved independently.
- Data should be shared with other software systems in the organization.

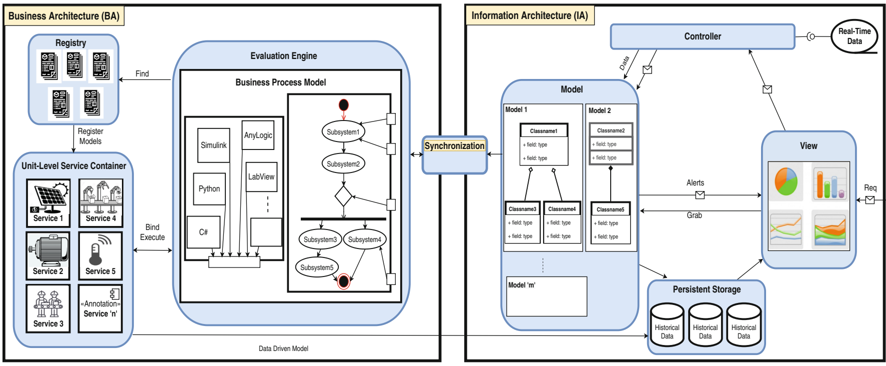

Fig. 2. The EA Blueprint Pattern

Solution: The EA Blueprint Pattern builds on EA concepts [17], and consists of three macro components (see Fig. 2): (1) *Business Architecture* (BA) represents business processes and interactions at organization level, (2) *Information Architecture* (IA) provides the technology infrastructure, the business application logic, and the data representing the current status of the organization, and (3) *Synchronization* handles the information exchange between BA and IA.

Resulting Context: Stable and flexible architecture, as new DTO elements can be easily added as services. Services are cohesive and loosely coupled.

Related Patterns: Service Oriented Architecture, Model View Controller.

3.1 Business Architecture (BA)

The *Business Architecture* is well positioned to play a key role in realizing a DTO, as it embeds *models* and *processes* that should be mimicked by the DTO. Since the organization is a dynamic entity continuously evolving in response to external forces (e.g., market trends, macro-economics) [2], its DTO should be easily evolved. To this end, EA Blueprint Pattern promotes the exploitation of Service-Oriented composition/decomposition style and Agile Design principles [10] to implement the BA. Indeed, such a combination improves DTO *modularity*, allows for different levels of *granularity*, and facilitates evolution [5].

In particular, the BA consists of three architectural elements: *Unit-Level Service Container*, *Registry*, and *Evaluation Engine* (see left-end side in Fig. 2).

The *Unit-Level Service Container* is in charge of managing the deployment and provision of organization services in DTO. Unit-Level Services serve as digital representations (models) of organizational elements – such as machines, spaces, activities, buildings, software and people – and provide their actual status and behavior. In Sect. 2.1, for the Hospital example, each unit that gives service is considered as a *Unit-Level Service*. For example, *Unit-Level Service Container* includes emergency unit, brain unit, surgery unit, cardiology or coronary care unit, etc. These services have some attributes like their capacity, IDs, certain equipment, and so on. According to SOA, the unit-level services are registered in the service *Registry*. Therefore, when elements in the physical organization change or are added/removed, their representation in *Unit-Level Service Container* should also be changed or added/removed accordingly. Further, unit-level services can be composed to create complex functionality and increase the level of abstraction.

As some services might need to perform complex analysis on *Historical Data* (e.g., computing trends or possible futures), they can directly access the *Persistent Storage* in IA and retrieve the data of interest.

On the other hand, the *Evaluation Engine* is in charge of orchestrating the unit-level services to implement the DTO behavior, as specified by the *Business Process Model*.

Since the organization is in continuous change, both the unit-level services and the business processes may need frequent evolution. Propagating the changes at the unit-level to allow the *Evaluation Engine* to always invoke the latest service version may overwhelm DTO maintainers. Moreover, not all changes in the organization are completely disruptive for the DTO. On the contrary, some parts of the Business Process Model can continue working acceptably well after an evolutionary change that modifies the DTO required functionalities. In these cases, even if the BA degrades its utility because it uses partially aligned versions of the unit-level services, it still provides some functionality to the organization.

For these reasons, the unit-level services are registered in the *Registry* including their version, so that *Evaluation Engine* can find and dynamically bind to the most appropriate version.

3.2 Information Architecture (IA)

The Information Architecture macro-component is in charge of: (*i*) handling the data streams from organization elements, collecting the raw data, and creating the information needed at the business level, (*ii*) handling the persistence of historical data, (*iii*) and visualizing the information of interest.

To this end, EA Blueprint Pattern leverages Model-View-Controller (MVC), as it is a suitable pattern for providing the aforementioned functionalities (see right-end side of Fig. 2). However, other patterns may also be considered as alternatives for implementing this macro component.

The *Controller* entity handles the real-time data streams from organizational elements (e.g., sensors, IoT devices, cameras, IT applications, etc.) embedded in the organization. Incoming data are either Time-based or Event-driven. Time-based data streams pass to the controller with a specific sample time. Event-driven data are received by occurring an event in the physical world while carrying important information about the event. The *Controller* processes data and feeds them into the *Model*. *Controller* is also responsible for manipulating the data according to the common representation defined by the *Model*.

Consequently, the controller is the first entity in providing resiliency regarding the Operational Changes. Data transmission problems are frequent when working with data streams from multitude of sources and IoT devices. For example, when a sensor is replaced by a new sensor sending data in a different format, the *Controller* needs to be updated to produce meaningful updates in the *Model*. In addition, hardware failures, connection loss, low battery, electromagnetic noise, or human errors providing inputs or disconnecting devices are potential problems lead to Operational or Evolution changes. Therefore, a combination of graceful degradability and recoverability may suffice for the *Controller*, but robustness to changes would also be optimal and achievable in some cases.

The *Model* represents the (real-time) information needed at the organization level. Indeed, the information retained by the *Model* depends on the semantics of the data streams, but is agnostic to the specific format. The *Model* is also responsible for sending the data to the *Persistent Storage*. Due to security issues, it is recommended that the *Model* establishes as few communications as possible with the persistent storage.

The *View* offers GUI for visualizing the organization status and other information of interest.

3.3 Synchronization

In order to fully transfer an organization into a DTO, a key requirement for the BA is to be *aligned* with the supporting IA. The *Synchronization* element in EA Blueprint Pattern is responsible for feeding BA with real-time information from the IA.

The *Synchronization* element is necessary to use the DTO beyond the evaluation (e.g. using data analytics, simulation, machine learning) of possible future scenarios using historical data. These evaluations focus on what could happen in the organization (what-if scenario), but not on what is currently happening [8]. However, constant change can quickly make the evaluation outdated and the manual task of re-configuring an evaluation technique to incorporate recent changes makes it cumbersome. Therefore, EA Blueprint Pattern includes a first-class element that deals with the synchronization of real-time information at business level.

Despite the apparent conceptual simplicity for this element, a recent study of methods concerning the connection of real-time data with their related models demonstrates that the majority of methods in this field are non-standard [11].

In EA Blueprint Pattern, the *Business Process Model* at the BA accepts external information through ports, which are designed to inject into the *Business Process Model* specific real-time data retrieved from the *Model*. In this regard, two important steps should be performed for synchronizing the data: (1) identifying the two specific points in the BA and IA that must be aligned, and (2) effectively and efficiently feeding the BA with real-time information. The first step could be done based on knowledge of the organization by identifying the digital IDs of different organizational elements. For the second step, the *Synchronization* entity converts the received data to interpretable format for models in *Evaluation Engine*. Then, depending on data types, time-based or event-driven, sends the received data to the associated device model recognized in the first step.

Synchronization entity is also the second main actor to handle *Operational Changes*. For doing this, it needs the support of a dependable Controller in the IA since it relies on the information created by the controller when the *Operational Change* happened. Since the BA needs meaningful information in order to perform its tasks, the synchronization component should take special care on synchronizing the BA with information that was recorded in the IA when a disturbance happen in a data stream.

4 Related Work

Literature related to this work is manifold and spans over different topics, from Software Architecture and Architectural Patterns to Digital Twin Architecture. While Software Architecture and Architectural Patterns have been extensively investigated [13], Digital Twin Architecture is a relatively new field. Therefore, we summarise hereafter the state-of-the-art in Digital Twin Architectures.

Josifovska et al. [6] propose a reference framework for developing Digital Twins of cyber-physical systems organized according to the five-level architecture. In particular, the Digital Twin framework is structured around four main building blocks including Physical Entity Platform, Virtual Entity Platform, Data Management Platform, and Service Platform.

Generic Digital Twin Architecture (GDTA) relies on 5D-Digital Twin Model [19] and align with the information technology layers of the Reference Architecture Model Industry 4.0. GDTA is composed of five layers including: Asset Layer—The physical, Integration Layer, Communication Layer, Information Layer, Functional Layer, and Business Layer. In particular, the Functional Layer represents the core of this architecture and consists of five different sections: Simulation Services, Monitoring Services, Diagnosis Services, Prediction Services, Control Services, and Reconfiguration Services.

Redelinghuys et al. [12] propose a six-layer architecture as a solution for a specific manufacturing case study. The architecture includes an IoT Gateway layer to transfer information between the physical and virtual space, a cloud-based data storage, and an emulation and simulation layer.

Talkhestani et al. [18] describe an architecture of a Digital Twin in an intelligent automation system. In particular, the proposed architecture consists of two parts: a Digital Twin, and an Intelligent Digital Twin. The Digital Twin architecture includes models and interfaces to related modeling tools, models' version management, operational, organizational and technical data, relations to other DTs and to the real world. Whereas, the Intelligent Digital Twin is one level higher and enables for optimization of the process flow, automatic control code generation for newly added machines, and predictive maintenance.

Malakuti et al. [9] identify, as a result of interviews with different experts and reviews of existing proposals, nine requirements that should be fulfilled by Digital Twins. In order to fulfill such requirements, they introduce an abstract layered architecture for building digital twins and synthesizing the information from different sources.

Alam et al. [1] propose a digital twin architecture reference model for cloud-based cyber-physical systems (C2PS). The suggested architecture includes five layers. First and second layers have been allocated to physical and cyber entities, respectively, where every physical entity is straightforwardly mapped to a digital representative hosted in the cloud. A one-to-one connection between digital and physical entities is assumed for synchronizing the state of physical objects and their digital representative in the cloud. The third layer is responsible for communication and networking, whereas the fourth layer acts as a middleware managing digital entities, their active relations, and the related ontologies. Finally, the last layer is in charge of managing the visibility and privacy of the entities as well as of handling the data consumption and visualization from stakeholders.

All the aforementioned works provide general solutions, at different level of abstractions, pointing out the need for having an architectural entity (e.g., layer, component) specifically dedicated to DTO. However, none of them detail how to develop such a specific entity. Also, discussing on the need for resilient DTO is the missing part of these works. On the other hand, several papers leverage DTO to reach an organization or system resiliency like [15] which is not in the scope of this paper.

5 Conclusion

A Resilient Digital Twin of the Organization (DTO) is capable to keep or recover its utility and functionality in confronting different types of organizational changes. Although developing a Resilient DTO is a grand challenge, it facilitates actualizing of a resilient organization. Therefore, the current paper has investigated how to develop a resilient DTO to evolve according to the organization's needs. As the result, EA Blueprint Pattern has been proposed. This pattern, leveraging Enterprise Architecture, offers an architectural pattern that consists of three Macro components: (*i*) Business Architecture (BA), (*ii*) Information Architecture (IA), and (*iii*) Synchronization element. BA mimics the business processes and interactions at organization level whereas IA is responsible for managing real-time data. Synchronization handles the information exchange between BA and IA.

Acknowledgements. This research is funded by the Swedish Knowledge Foundation, Grant No. 20200117: ALADINO – ALigning Architectures for DIgital twiN of the Organization.

References

1. Alam, K.M., El Saddik, A.: C2PS: a digital twin architecture reference model for the cloud-based cyber-physical systems. IEEE Access **5**, 2050–2062 (2017)
2. Andersson, J., Caporuscio, M.: Aligning architectures for sustainability. In: Proceedings of the 10th European Conference on Software Architecture Workshops, ECSAW 2016 (2016)
3. Andersson, J., Grassi, V., Mirandola, R., Perez-Palacin, D.: A distilled characterization of resilience and its embraced properties based on state-spaces. In: Calinescu, R., Di Giandomenico, F. (eds.) SERENE 2019. LNCS, vol. 11732, pp. 11–25. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-30856-8_2
4. Andersson, J., Grassi, V., Mirandola, R., Perez-Palacin, D.: A conceptual framework for resilience: fundamental definitions, strategies and metrics. Computing **103**(4), 559–588 (2020). https://doi.org/10.1007/s00607-020-00874-x
5. Caporuscio, M., Edrisi, F., Hallberg, M., Johannesson, A., Kopf, C., Perez-Palacin, D.: Architectural concerns for digital twin of the organization. In: Jansen, A., Malavolta, I., Muccini, H., Ozkaya, I., Zimmermann, O. (eds.) ECSA 2020. LNCS, vol. 12292, pp. 265–280. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-58923-3_18
6. Josifovska, K., Yigitbas, E., Engels, G.: Reference framework for digital twins within cyber-physical systems. In: 2019 IEEE/ACM 5th International Workshop on Software Engineering for Smart Cyber-Physical Systems, pp. 25–31 (2019)
7. Laprie, J.C.: From dependability to resilience. In: DSN 2008 (2008)
8. Lu, Y., Liu, C., Wang, K.I.K., Huang, H., Xu, X.: Digital twin-driven smart manufacturing: connotation, reference model, applications and research issues. Robot. Comput.-Integr. Manuf. **61**, 101837 (2020)
9. Malakuti, S., Schmitt, J., Platenius-Mohr, M., Grüner, S., Gitzel, R., Bihani, P.: A four-layer architecture pattern for constructing and managing digital twins. In: Bures, T., Duchien, L., Inverardi, P. (eds.) ECSA 2019. LNCS, vol. 11681, pp. 231–246. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-29983-5_16
10. Martin, R.C.: Agile Software Development: Principles, Patterns, and Practices. Prentice Hall PTR, Hoboken (2003)
11. Modonia, G.E., Caldarolaa, E.G., Saccob, M., Terkajb, W.: Synchronizing physical and digital factory: benefits and technical challenges. Procedia CIRP **79**, 472–477 (2019)
12. Redelinghuys, A.J.H., Basson, A.H., Kruger, K.: A six-layer architecture for the digital twin: a manufacturing case study implementation. J. Intell. Manuf. **31**(6), 1383–1402 (2019). https://doi.org/10.1007/s10845-019-01516-6
13. Richards, M.: Software Architecture Patterns. O'Reilly Media Inc., Sebastopol (2015)
14. Rosen, R., von Wichert, G., Lo, G., Bettenhausen, K.D.: About the importance of autonomy and digital twins for the future of manufacturing. IFAC-PapersOnLine **48**(3), 567–572 (2015). 15th IFAC Symposium on Information Control Problems in Manufacturing
15. Saad, A., Faddel, S., Youssef, T., Mohammed, O.A.: On the implementation of IoT-based digital twin for networked microgrids resiliency against cyber attacks. IEEE Trans. Smart Grid **11**(6), 5138–5150 (2020)
16. Schmeck, H., Müller-Schloer, C., Çakar, E., Mnif, M., Richter, U.: Adaptivity and self-organization in organic computing systems. ACM Trans. Auton. Adapt. Syst. **5**(3), 10:1–10:32 (2010)

17. Sousa, P., Caetano, A., Vasconcelos, A., Pereira, C., Tribolet, J.: Enterprise architecture modeling with the unified modeling language. In: Rittgen, P. (ed.) Enterprise Modeling and Computing with UML, pp. 67–94. IGI Global (2007)
18. Talkhestani, B.A., et al.: An architecture of an intelligent digital twin in a cyber-physical production system. at - Automatisierungstechnik **67**(9), 762–782 (2019)
19. Tao, F., Zhang, M., Nee, A.: Five-dimension digital twin modeling and its key technologies. In: Digital Twin Driven Smart Manufacturing, pp. 63–81. Academic Press (2019)
20. Watts, S.: Digital twins and the digital twin of an organization (DTO). https://www.bmc.com/blogs/digital-twins
21. Woods, D.D.: Four concepts for resilience and the implications for the future of resilience engineering. Reliab. Eng. Syst. Saf. **141**, 5–9 (2015)