A Conceptual Architecture for Building Digital Twins

Paula Muñoz, Javier Troya and Antonio Vallecillo

ITIS Software. Universidad de Málaga, Spain

Abstract

Digital twins have recently emerged as an innovative paradigm for improving the performance of cyber-physical systems, software and processes. In essence, a digital twin is a virtual replica of the system whose operation is to be optimized by improving its performance, detecting possible anomalies, or enabling its preventive maintenance. Although some consensus exists in the community about what a digital twin is, there are still unresolved questions about its definition, architecture, and related concepts. This paper proposes an architecture for the implementation of digital twins, and compares it with existing proposals in the literature with the goal of reaching a common understanding.

Keywords

Digital twins, Data Lake, Architecture

1. Introduction

Digital Twins have proven to be powerful mechanisms for optimizing the performance of various types of cyber-physical systems, processes, and software services [1, 2]. Their underlying idea is the *replication* of an existing system. The earliest applications of this concept can be traced back to NASA's Apollo mission [3], where engineers created replicas of the systems sent into space to assist astronauts from Earth by emulating the behavior of those systems and enabling what-if analyses. These Digital Twins allowed ground engineers to evaluate optimal actions using the replicas, ensuring the astronauts' safety and preserving the integrity of the spacecraft.

In early applications, replicas were physical and synchronization was manually achieved. However, once the potential value and benefits of these techniques was understood, replicas were digitized using simulation or analytical models, e.g., differential equations, and the synchronization between them was automated. In our work, we will refer to the latter type of model as Digital Twin (DT). According to the Digital Twin Consortium, a DT is "a digital replica of a system that is synchronized with a certain frequency and with a certain level of fidelity." In the literature, the replicated system is referred to as a *Physical Twin* (PT), or *Observable Object* according to the ISO standard on Manufacturing Digital Twins [4].

In this paper, we present a modular architecture that introduces the concept of a *Digital Twin System* (DTS). The DTS includes not only the physical system, its replica, and the services associated with the DT, but also the connections between the two twins. The system's replica, i.e., the DT, can take various forms, such as algorithms, simulation models, or analytical models defined in terms of differential equations. It should be noted that a simulation model or an algorithm alone cannot be considered a DT. These elements are only considered as such within the context of a DTS, where there is an automatic and bidirectional synchronization with the replicated system (the PT) and where data and commands are exchanged between them to optimize the system's performance.

In this work, we want to precisely define the different concepts that comprise any DTS. The current terminology is not clear enough to distinguish between the digital replica (what we call DT) and the complete system itself (the DTS), which also involves the physical system, the data exchange mechanisms, and the associated services. Very often the term "Digital Twin" is used in both cases, which is confusing. Our work advocates a symmetric definition in which we have two twin systems whose operations are always kept synchronized. The remaining elements of the architecture serve to ensure the correct interaction between the two twins and to optimize the operation of the target system based on its DT. This complete system is what we call DTS in order to clearly distinguish it from the digital replica.

The proposed architecture also aims to optimize the scalability and composability of DTs. To achieve this, the architecture is centered around a *Data Lake* (DL) [5], i.e., a centralized repository designed to store, process, and secure large amounts of structured, semistructured, and unstructured data. A set of drivers access the DL and orchestrate the exchange of data and commands between the elements of the DTS.

This paper defines the architecture and its main elements, and compares it with other conceptual architectures commonly used in the literature.

MESS@STAF 2023: International workshop on MDE for Smart IoT Systems, July 18–21, 2023, Leicester, United Kingdom
paulam@uma.es (P. Muñoz); jtroya@uma.es (J. Troya); av@uma.es (A. Vallecillo)
0000-0003-2939-5803 (P. Muñoz); 0000-0002-1314-9694 (J. Troya); 0000-0002-8139-9986 (A. Vallecillo)

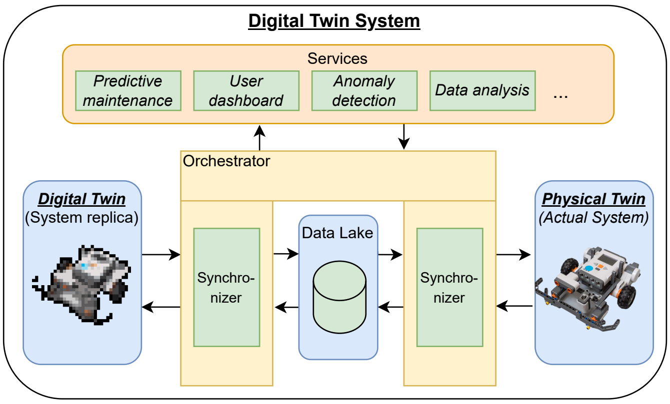

Figure 1: Our proposed Digital Twin System Architecture.

2. Digital Twin (System) Architecture

Figure 1 depicts our proposed architecture for the implementation of Digital Twin Systems (DTSs). Before providing a detailed description, it is essential to define several key terms:

- **Physical Twin** (also **Actual System** or **Observable Object**). This is the system, service, or product whose behavior we want to optimize using a digital replica. The state of the PT is continuously monitored, and the values of its relevant properties are periodically obtained. The PT may already exist when the DT is created, or it may not exist, for example, if we intend to use the DT during the design phase of the physical twin—that is, whether the DT models are *scientific* or *engineering* models [6].
- **Digital Twin**. It is the synchronized replica of the PT, updated at specific intervals and modeled with the required fidelity level to capture the system's properties of interest at the necessary level of detail. The DT represents the system's properties of interest and emulates its behavior using data and (analytical or simulation) models, which are continuously updated throughout the system's lifespan and stored in the DL.

Building upon the above definitions, we define a **Digital Twin System (DTS)** as a system engineered for a specific purpose, composed of:

- The *Physical Twin* (PT).
- The virtual replica, i.e., the *Digital Twin* (DT).
- The synchronization mechanisms that facilitate the exchange of data and commands between the twins.
- A set of *Services* that enable the exploitation of the data produced and exchanged between the twins to help achieve the DTS objectives, e.g., performance improvement, predictive maintenance, or anomaly detection.

To achieve maximum decoupling among the elements and ensure scalability and flexibility, our architecture connects the elements through a Data Lake, implementing the Blackboard architectural pattern [7]. This allows all elements to asynchronously exchange information by reading from and writing to the DL. If there is critical information that requires minimal latency, elements can also connect directly through the *Orchestration* component. This component facilitates both direct exchange of information between elements and connection to the database. Additionally, the Orchestration component includes the necessary synchronization services to ensure the correct operation of the system.

One of the significant advantages of this architecture is its simplicity in connecting the elements, and the highly decoupled connection between them. Adding a new element would only require adapting the information input to the Orchestration component without affecting any other system elements. Similarly, if multiple Digital Twins need to be interconnected and their behavior synchronized within the DTS, it can also be achieved through the Orchestration component. We can also have several Digital Twins with different levels of fidelity organized to choose the best one to use at each moment depending on the state of the PT, the concrete performance requirements, or the degree of faithfulness needed. Additionally, we can replace the PT with another virtual model for comprehensive system testing. All these changes can be accomplished by simply modifying the specific interface implementation for the Orchestration component.

This architecture has been validated through the successful application in the development of two specific Digital Twin Systems: a Lego Mindstorms NXT car [8] and an Arduino robotic arm [9].

3. Current Proposals

The initial conceptualization of an architecture for Digital Twins was originally proposed by Grieves [3], and consisted of three elements: the DT, the PT and the connections between them. This conceptual architecture did not yet include the concept of *services* as an external element to the Digital Twin nor the need for a data store for the system. This architecture resulted from NASA's experiments, where this new technology relied on the existence of a virtual system that replicated the physical system and was synchronized in real-time.

Subsequent architectures for digital twin systems expand this initial architecture by incorporating additional elements but retaining the core idea of replicating the actual system. For instance, the architectures based on five elements [10, 11, 12] include the physical entity, the virtual entity, a database, a set of services, and the connections between them. In other proposals such as [10, 11], the DT is defined as the combination of all the elements, rather than just the digital replica—the opposite of what the Digital Twin Consortium proposes [1], and we adopt in our proposal. In [12], the Digital Twin is separated from the Data Lake and the connections, but it includes the optimization services—thus hindering the addition of new services or the modular modification of existing ones. The concept of PT is not considered in [12], either.

The architecture defined by ISO for Digital Twins for Manufacturing [4] is similar to these recent proposals, as well as to ours: they introduce the concept of *Digital Twin Framework*, which corresponds to our Digital Twin System (DTS), but excluding the physical system.

A systematic mapping study that provides a comprehensive overview of software architectural solutions for DTs is presented in [13]. In addition to the most representative architectures, such as those described above, the paper describes many architectures for specific systems and uses a two-dimensional classification of architectures [14], based on their levels of abstraction and detail, to classify them. A useful catalog of 14 quality attributes relevant to DTs is also proposed.

The authors of [15] propose the use of Domain-Driven Design [16] to tackle the design of architectures for DTSs, instead of implementing pre-defined reference architectures, such as the five elements one [10]. The analysis of the possible advantages and disadvantages of designing a new architecture following a method such as DDD versus the implementation and deployment of an existing reference architecture represents an interesting research challenge.

Finally, another concept used in the literature about Digital Twins is that of Digital Shadow (DS). This term is defined as the set of system data traces and their aggregation, collected for a specific purpose. These traces in our system are stored in the Data Lake and exploited by the various services. The main difference between a simulation model, the DS and the DT is their degree of synchronization and interaction with the PT. The first is neither synchronized with the PT nor interacts with it. The second is permanently updated with data collected from the PT, but does not send information back to the PT. The DT remains synchronized with the PT and interacts with it during its lifetime [17].

An initial version of the architecture proposed here was presented in [8]. The main difference with this one is the inclusion of the Orchestrator component. It helps improve the connections among the different elements of the DTS, and enables the implementation of more efficient synchronization mechanisms between them.

To conclude, it is worth mentioning that our proposal emphasizes the natural symmetry that exists when talking about twins, two sister entities that are replicas of each other and that continuously synchronize and exchange both data and control operations between them, and with a set of services that try to exploit this information. Without such synchronization, one cannot speak of a Digital Twin. This is why we call them physical and digital twins. In addition, we place great emphasis on the decoupling between the elements of the architecture to allow maximum independence and scalability when composing them.

Further work includes validation through the development of more DTSs, the analysis of its properties (such as scalability, performance, or maintainability, among others), and the definition of mappings between its elements and those of other proposals to improve its interoperability and expand its use.

Acknowledgments

An initial version of this architecture was presented in [8], and since then it has been refined during discussions with many Digital Twin experts and practitioners. In particular, we would like to thank the participants of the 2022 Dagstuhl Seminar 22362 “Model-Driven Engineering of Digital Twins” [18] and of the 2023 Cargèse CAMPaM workshop on “Model-Based Systems Engineering for and by Digital Twins” for their insightful comments and suggestions, which allowed us to refine and improve the architecture presented in this paper. Special thanks to Andreas Wortmann for his useful pointers to existing architectures and for his wise comments. Thanks also to the reviewers of this paper for their valuable comments and suggestions.

This work was partially funded by the Spanish Government under projects PID2021- 125527NB-I00 and TED2021-130523B-I00, as well as by the University of Málaga.

References

[1] Digital Twin Consortium, Glossary of digital twins, https://www.digitaltwinconsortium.org/glossary/index.htm, 2021.

[2] M. Dalibor, N. Jansen, B. Rumpe, D. Schmalzing, L. Wachtmeister, M. Wimmer, A. Wortmann, A cross-domain systematic mapping study on software engineering for digital twins, J. Syst. Softw. 193 (2022) 111361.

[3] M. Grieves, J. Vickers, Digital Twin: Mitigating Unpredictable, Undesirable Emergent Behavior in Complex Systems, Springer, 2017, pp. 85–113.

[4] ISO 23247:2021, Automation systems and integration – Digital Twin Framework for manufacturing, International Organization for Standardization (ISO), 2021.

[5] R. Hai, C. Quix, M. Jarke, Data lake concept and systems: a survey, CoRR abs/2106.09592 (2021). arXiv:2106.09592.

[6] E. A. Lee, M. Sirjani, What good are models?, in: Proc. of FACS 2018, volume 11222 of *LNCS*, Springer, 2018, pp. 3–31. doi:10.1007/978-3-030-02146-7_1.

[7] F. Buschmann, R. Meunier, H. Rohnert, P. Sommerlad, M. Stal, Pattern–Oriented Software Architecture: A System of Patterns, John Wiley & Sons, 1996.

[8] P. Muñoz, J. Troya, A. Vallecillo, Using UML and OCL Models to Realize High-Level Digital Twins, in: Proc. of ModDiT@MODELS'21, IEEE, 2021, pp. 212–220.

[9] D. Pérez-Porras, P. Muñoz, J. Troya, A. Vallecillo, Key-Value vs Graph-based data lakes for realizing Digital Twin systems, in: Proc. of MeSS@STAF'22, volume 3250 of *CEUR Workshop Proceedings*, 2022. URL: https://ceur-ws.org/Vol-3250/messpaper2.pdf.

[10] F. Tao, M. Zhang, Y. Liu, A. Nee, Digital twin driven prognostics and health management for complex equipment, CIRP Annals 67 (2018) 169–172.

[11] F. Tao, H. Zhang, A. Liu, A. Y. C. Nee, Digital twin in industry: State-of-the-art, IEEE Trans. Ind. Informatics 15 (2019) 2405–2415.

[12] M. Dalibor, J. Michael, B. Rumpe, S. Varga, A. Wortmann, Towards a model-driven architecture for interactive digital twin cockpits, in: Proc. of ER'20, volume 12400 of *LNCS*, Springer, 2020, pp. 377–387.

[13] E. Ferko, A. Bucaioni, M. Behnam, Architecting digital twins, IEEE Access 10 (2022) 50335–50350. doi:10.1109/ACCESS.2022.3172964.

[14] M. Broy, M. Gleirscher, P. Kluge, W. Krenzer, S. Merenda, D. Wild, Automotive Architecture Framework: Towards a Holistic and Standardised System Architecture Description, Technical Report TUM-I0915, IBM Corporation, Technische Universität München, Germany, 2009.

[15] A. Macías, E. Navarro, C. E. Cuesta, U. Zdun, Architecting digital twins using a domain-driven design-based approach, in: Proc. of ICSA'23, IEEE, 2023, pp. 153–163. doi:10.1109/ICSA56044.2023.00022.

[16] E. J. Evans, Domain-driven design: Tackling complexity in the heart of software, Addison-Wesley, 2004.

[17] M. Dalibor, M. Heithoff, J. Michael, L. Netz, J. Pfeiffer, B. Rumpe, S. Varga, A. Wortmann, Generating customized low-code development platforms for digital twins, J. Comput. Lang. 70 (2022) 101117.

[18] L. Cleophas, T. Godfrey, D. E. Khelladi, D. Lehner, B. Combemale, B. Rumpe, S. Zschaler, Model-Driven Engineering of Digital Twins (Dagstuhl Seminar 22362), Dagstuhl Reports 12 (2023) 20–40. URL: https://drops.dagstuhl.de/opus/volltexte/2023/17808.