ISSN 2723-6471

Synchronization Patterns for Digital Twin Systems

Wael Alghamdi[1,*], Emad Albassam[2,]

[1,2]*Department of Computer Science, Faculty of Computing and Information Technology, King Abdulaziz University, Jeddah, Saudi Arabia*

(Received: June 12, 2024; Revised: June 30, 2024; Accepted: July 11, 2024; Available online: July 23, 2024)

Abstract

In the area of digital twin technologies, ensuring operational accuracy and efficiency requires information alignment between the physical objects and their corresponding digital twins. This study presents a catalog of synchronization patterns that can be incorporated into the design of digital twin systems to help improve data quality and alignment in these systems. The catalog consists of 11 synchronization patterns and was developed based on analyzing prior synchronization solutions in digital twin systems. This catalog provides an overview of each pattern, its applicability, and its advantages and limitations. With this approach, it is possible to study the benefits and limitations of various synchronization patterns, such as time-driven, event-driven, hybrid, and adaptive patterns, and determine their applicability in different operational settings. Identified patterns are simulated in several component-based software architectures, where synchronization patterns are embedded in components' connectors to minimize modification in these components. The study results indicate that it is crucial to identify and incorporate appropriate synchronization patterns in the design of digital systems to maximize the benefit of this technology. While some patterns work well in industrial settings, others are more applicable in domains such as health systems and smart cities. The findings offer valuable directions for future innovations and use in various industries, thereby significantly raising the field of digital twin technology.

Keywords: Digital Twins Technology, Synchronization, Patterns, Synchronization Triggers, Software Design

1. Introduction

Several areas have been revolutionized with the digital twin's technology, in which virtual replicas are created from physical objects such as physical entities, processes, and locations [1], [2], [3]. Synchronization is a crucial attribute of digital twin systems. It ensures that a digital twin adequately reflects the state of its physical object through continuous alignment between a physical object and its digital twin, providing a reliable digital twin for command, diagnostics, and forecasting maintenance, as shown in figure 1. The overall added value obtained from this technology in these systems is affected by proper alignments between twins via synchronization since the synchronization process replicates a real-world object, enabling it to support better decision-making and predictive maintenance. This is equally critical to organizations, as live data and predictive analytics have the potential to influence operational outcomes greatly.

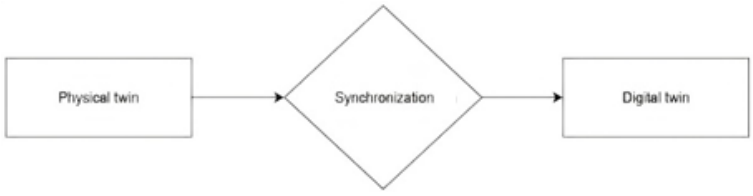

Figure 1. Overview of the Concept of Synchronization in Digital Twins

The synchronization concept in digital twin systems has grown from basic data mirroring to a complex, real-time interaction between physical objects and their digital twins. Synchronization in digital twin systems can involve (a) data synchronization, (b) process synchronization, and (c) time synchronization. In data synchronization, the synchronization mechanism must transmit sensor data in real time and ensure its consistency and integrity during network outages or latency. Among the issues that data synchronization must handle on a large scale, it is possible to mention vast volumes of data, quality preservation consideration, and heterogeneities of different formats. In process

***Corresponding author: Wael Alghamdi (walghamdi0419@stu.kau.edu.sa)**
DOI: https://doi.org/10.47738/jads.v5i3.267

ISSN 2723-6471

synchronization, synchronization covers process alignment, in which the digital twin's analytical or simulation activities are correlated with actual subroutines wherein an asset operates. It is essential to ensure that the information obtained from using digital twin systems remains meaningful and up-to-date. Time Synchronization aligns each digital twin state and temporalized state's status with the physical object. This becomes critical for real-time systems such as production or logistics automation.

Synchronization in digital twin systems can be achieved through software-based methods, in which various software tools, including protocols for real-time communication, middleware solutions, or reliable APIs, are used. These tools must also support huge volumes and varied data types and guarantee reliable transmission. Synchronization also requires hardware considerations since hardware is a major factor that significantly influences synchronization. This includes sensor placement and deployment, network architecture for conveying data, and computational resources for data processing and analysis.

Several challenges need to be addressed to utilize the digital twin technology effectively, including (1) real-time data handling addressing concerns such as high throughput, low latency, and data consistency, even when losses or transmission mistakes could occur, (2) privacy concerns of synchronized data [4], (3) the complexity of systems since digital twins may integrate multiple (sub)systems and components, all of which have their own needs and qualities, which increases the overall complexity, and (4) precision and reliability since, in many domains, precise and constant synchronization is required. This is particularly important in highly critical domains, such as aerospace or medicine, as incorrect synchronization may lead to catastrophic results.

Furthermore, the emergence of digital twin technology in variant domains such as the manufacturing, health care, automotive, and energy industries indicates a necessity for understanding various synchronization techniques, where the applicability of these techniques depends on operational conditions and technological constraints, all while managing data complexity and volume.

Considering these difficulties, this study investigates synchronization techniques commonly incorporated in the design of digital twin systems. We describe these techniques as synchronization patterns recurring in specific contexts and settings. This approach allows practitioners to understand the applicability of these patterns and evaluate their advantages and disadvantages according to business needs. The research questions of this work are as follows:

RQ1: What real-time synchronization mechanisms exist to achieve data alignment in digital twins?

RQ2: What synchronization patterns can be applied in digital twin systems that work best under specific contexts/settings/scenarios?

To address RQ1, this paper first provides the results of a comprehensive literature review of various synchronization approaches reported in prior works in digital twin systems. For RQ2, the paper presents a catalog of 11 synchronization patterns developed based on careful analysis of the results obtained from the literature review. Several research efforts were made in the past, notably in event-driven synchronization in industrial scenarios and time-driven techniques for environmental monitoring and hybrid schemes on smart city infrastructures, which have achieved efficient operations, accurate data generation, and prompt system responsiveness. However, these studies also highlight the importance of more flexible and integrated synchronization approaches, especially in multi-tiered system environments, which are large-scale systems operating within volatile contexts. Therefore, this paper seeks to fill existing gaps by giving a holistic view of patterns for synchronizing digital twins that apply to different use case scenarios. This paper's findings are expected to enhance the effectiveness of digital twin systems.

2. Literature Review

To address RQ1, we conducted a comprehensive literature review of prior works on synchronization in digital twin systems. The literature review was conducted by searching various scholarly databases, including IEEE Xplore, Google Scholar, ResearchGate, and ScienceDirect. Our search terms consist of "digital twin" and "synchronization" to cover the full range of prior works related to these terms. The results were refined further by reviewing and evaluating the abstract section to determine the relevance of such works. The results were then re-evaluated by reviewing the full text to assess its relevance.

 ISSN 2723-6471

Our results (see table 1) show that the research landscape in digital twin systems has offered a wide range of efforts, addressing diverse issues and problems related to the design and implementation of synchronization approaches. Based on a set of characteristics, prior works can be classified as follows.

Technological Frameworks: Various concepts have been considered in the research for digital twin synchronization. For example, prior works suggested the development of a blockchain-based digital twin (DT) framework to secure cyber-physical systems. This indicates that there have been attempts to merge emerging technologies such as Blockchain for better security and trust levels within Digital Twin models [5]. In [6], the authors provide a holistic framework and an analysis of the problem of digital twin synchronization, forming a foundational understanding regarding challenges and solutions in this area.

Application-Specific Studies: some prior works are tailored to specific digital twin applications, such as studies on digital twins for urban logistics, capturing the typical ways digital twin technology may be used in this particular area of practice [7].

Broader Technological Challenges and Directions: A few studies have provided a more comprehensive view of enabling technologies, challenges, and open research areas in digital twins, such as those involving AI and IoT integration. Thus, an emphasis is placed on combining several advanced emerging technologies under development in digital twins [8]. In [9], the authors provide a comparison framework for digital twin research to critical analysis for several techniques, trends, and future directions necessary for better understanding the overall trajectory of development regarding digital twins.

Therefore, despite a great deal of study, there are still many gaps in digital twin systems, including (1) the need for integrated and holistic frameworks since complex digital twin settings require seamless integration of various sources and types of data, which require handling of heterogeneity and achieving efficiency, (2) consideration for scalability and flexibility, as studies of synchronization patterns that are scalable and flexible enough to handle requirements in real-time remain limited, and (3) consideration for cross-domain synchronization strategies, since applications across different domains may also be considered when exploring strategies for synchronization since each has its unique set of challenges and requirements.

With this respect, this paper addresses the second gap by investigating suitable synchronization patterns for digital twins. To better understand such synchronization mechanisms, our approach categorizes prior works according to several factors, including (1) solution type, where we classify the type of solution, such as architecture, framework, and pattern (2) applicability of the solution indicating whether the synchronization solution provided by the authors is domain-independent or domain-specific (and in which domain), (3) whether prior implementation knowledge about the internal software system in which digital twins operate is required for applying the proposed solution, (4) the validation approach by the authors to validate their proposed solution (e.g., using simulations, case studies, etc.), and (5) the synchronization triggers for triggering the synchronization process in the proposed solution.

Table 1 summarizes the results of the literature review. Notably, most reviewed works require prior knowledge about the design/implementation of the system in which the digital twins operate. In addition, we have discovered that several approaches share common triggers for the synchronization process. For example, many existing methods rely on hybrid and adaptive synchronization, while other synchronization triggers are less common. These synchronization triggers are explained in section 3. Finally, the reviewed synchronization solutions cover a range of domains and contexts, where the majority are domain-dependent.

3. Research and Discussion

After carefully analyzing the literature review results, we developed a catalog of 11 synchronization patterns that can be incorporated into the design of digital twin systems. This section provides details of each pattern, including an overview of its mechanics, applications, advantages, and limitations.

 ISSN 2723-6471

Table 1. Results of literature review classifying prior works on synchronization in digital twin systems.

Paper	Solution Type	Applicability	Implementation Knowledge required?	Validation Approach	**Synchronization** Trigger									
					State Change	Time Based	Hybrid	Adaptive	Multi-Level	Event-Chain	Data-Driven	Context-Aware	Multi-Modal	None
[1]	Improvements in Methodology	Industrial Plants	YES	Empirical			X	X						
[2]	Framework	Material Recognition	YES	Case Study, Empirical	X								X	
[3]	Design Pattern, Framework	Manufacturing	YES	Case Study	X								X	
[5]	Practical Implementation	Urban Traffic and Logistics	YES	Data-Driven Optimization Techniques			X	X						
[6]	Suggestions, Discussing	Manufacturing, Smart Cities, Healthcare	NA	None										X
[7]	Architecture, Scheme	Industrial\IoT	YES	Simulation		X		X						
[8]	Suggestions, Discussing	Intelligent manufacturing, Industry 4.0	NA	None										X
[9]	Practical Implementation	Manufacturing	YES	Case Study, Simulation			X				X			
[10]	Architecture, Design Pattern	Domain-Independent	YES	Case Study	X									
[11]	Architecture	Manufacturing	YES	Simulation			X							
[12]	Framework	Metaverse	YES	Simulation			X							
[13]	Framework	Libraries	YES	Simulation			X							
[14]	Framework	Manufacturing	YES	Proof			X	X						
[15]	Suggestions, Discussing	Wireless Sensor Networks	YES	None			X				X			
[17]	Framework	Manufacturing	YES	Case Study, Simulation				X						
[18]	Framework	Metaverse Virtual Services	YES	Simulation, Empirical				X						
[19]	Framework	Manufacturing	YES	Simulation				X						
[20]	Framework	Manufacturing	YES	Case Study				X			X	X		
[23]	Architecture	Manufacturing, Industrial\IoT	YES	Case Study					X					
[24]	Architecture	Domain-Independent	YES	Case Study, Simulation						X				
[25]	Framework	Manufacturing	YES	Case Study							X			
[26]	Ontology	Manufacturing	YES	Case Study								X		
[30]	Suggestions, Discussing	Domain-Independent	NA	None										X
[31]	Suggestions, Discussing	Domain-Independent	NA	None										X
[32]	Suggestions, Discussing	Power Systems	NA	None										X

3.1. Event-Driven Synchronization Pattern

Overview: In this pattern, synchronization between the physical object and its digital twin is triggered in response to events or state changes occurring at the physical object (see figure 2). For example, an unmanned arial vehicle might receive a command to change direction or destination from a controlling system, which causes a state change. Such events trigger synchronization under this pattern. *Mechanics*: To realize this pattern, input events such as sensor inputs and external alerts to the physical objects that may cause a state change are identified. Synchronization is then triggered for relevant events, ensuring that the digital twin updates are directly correlated with significant changes in the physical object [2], [3], [10]. *Applicability*: This pattern applies to systems with well-defined input events, states, and transitions between these states (e.g., via state machines), such as industrial automation and emergency response systems, in which events that trigger state change can also trigger synchronization. *Advantages and Limitations*: This pattern delivers accurate and timely updates to the digital twin when important events or state changes occur at the physical object.

 ISSN 2723-6471

However, accurately identifying relevant events or states can be challenging, especially for complex systems with vast amounts of events that may contain noise. In addition, the digital twin's state can become outdated if no such events occur at the physical object for some time.

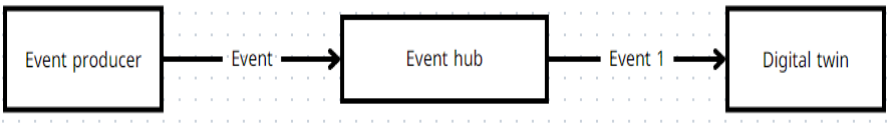

Figure 2. Event-Driven Pattern

3.2. Time-Driven Synchronization Pattern

Overview: As shown in figure 3, synchronization in this pattern is scheduled periodically based on predefined or fixed time intervals to ensure regular synchronization between the physical object and its digital twin. For example, the status of a physical server (e.g., CPU utilization) can be pulled regularly to trigger synchronization with the digital twin. *Mechanisms*: A timer helps update the state of the digital twin at regular intervals, irrespective of the current state of the physical object [7]. *Applicability*: This pattern is applicable to systems that require continuous monitoring, such as environmental sensing networks and health monitoring systems. *Advantages and Limitations*: synchronization between the physical object and the digital twin becomes steady. However, state changes in the physical twin that are only momentary can be missed. In addition, this pattern may result in unnecessary data transmissions during periods of inactivity at the physical object.

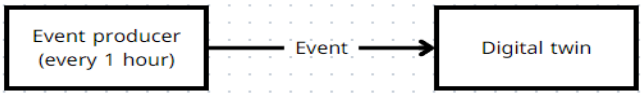

Figure 3. Time-Driven Pattern

3.3. Hybrid Synchronization Pattern

Overview: Combines event-driven and time-driven synchronization techniques to provide timely and steady updates for the digital twin in more demanding operational scenarios. As shown in figure 4, the state of the digital twin can be updated if either (1) the physical object receives an event or changes its state or (2) periodically at regular time intervals. *Mechanics*: This pattern transitions between event-based and time-driven modes based on whether input events exist at the physical object [1], [5], [9], [11], [12], [13], [14], [15]. *Applicability*: This pattern applies to physical objects that experience both periods of activity (by receiving input events) and inactivity (lack of input events), such as smart city management systems. *Advantages and Limitations*: This pattern provides accurate and timely Synchronization. However, this pattern can complicate the synchronization logic and system design.

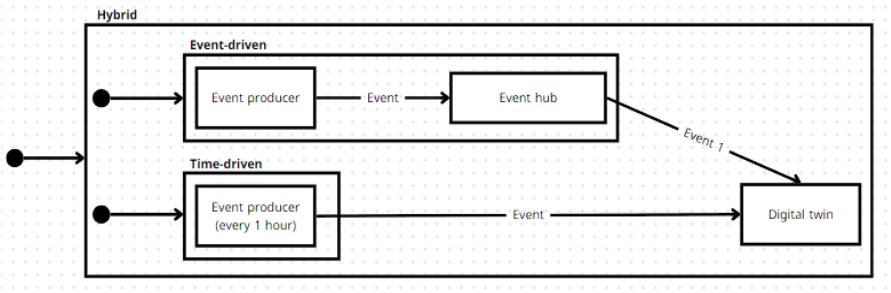

Figure 4. Hybrid Pattern

3.4. Adaptive Synchronization Pattern

Overview: This pattern dynamically adjusts the synchronization strategy based on real-time analysis of the performance, data flow, and network conditions at the physical object. As illustrated in figure 5, this pattern incorporates an adaptation algorithm that analyzes events in the physical object, such as analyzing the current traffic into the physical object. This pattern then adjusts the synchronization mechanism (e.g., the time interval of synchronization) according to predefined policies and thresholds. A close feedback loop, such as the MAPE-K for

 ISSN 2723-6471

autonomous systems [16], can be implemented to realize the adaptation algorithm. *Mechanisms*: this pattern utilizes algorithms that can learn and adapt over time to optimize synchronization by performing real-time analysis of system attributes. Such a pattern can incorporate a close feedback loop [1], [5], [7], [14], [17], [18], [19], [20] or an adaptive systems framework [21], [22]. *Applicability*: This pattern is applicable to systems with fluctuating operational conditions that change over time, such as adaptive traffic control systems that have to adjust to shifting traffic patterns. *Advantages and Limitations*: It provides high flexibility and system efficiency. Nevertheless, the underlying analysis algorithms may require frequent tuning and configuration due to their complexity.

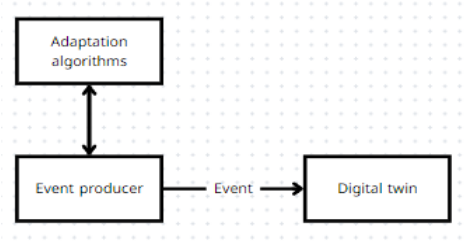

Figure 5. Adaptive Pattern

3.5. Multi-Level Synchronization Pattern

Overview: This pattern is illustrated in figure 6, in which several interconnected physical objects are present, such that a physical object at one layer may communicate with physical objects at other layers. In edge computing, for example, edge nodes are connected to multiple devices in the device layer. This pattern manages synchronization across multiple hierarchical levels within a system, and it is particularly relevant in complex systems comprising multiple interconnected digital twins. *Mechanics*: Manages synchronization across multiple hierarchical levels within a system, addressing the interdependencies and ensuring coherence between different levels of the digital twin [23]. *Applicability*: Large-scale and complex systems with multiple digital twins at different levels must operate harmoniously. *Advantages and Limitations*: Provides an overall view of the system via synchronization at all levels in a hierarchy. However, complexity increases significantly as the number of levels in the hierarchy and their interdependency increase.

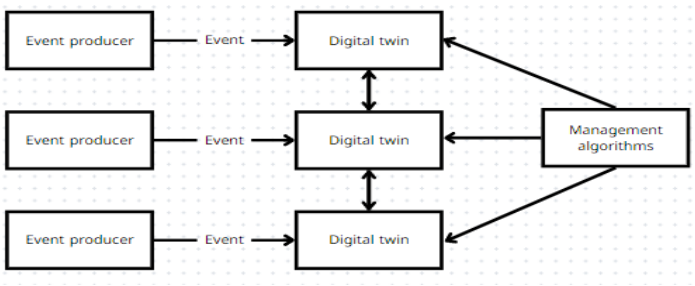

Figure 6. Multi-Level Pattern

3.6. Event-Chain Synchronization Pattern

Overview: This pattern focuses on the sequential and causal relationships of events within a system with significant interconnected dependencies. For example, in a distributed factory automation system, factory workstations can be connected sequentially, such that one workstation provides parts to the next workstation for processing. *Mechanics*: This pattern maintains alignment across multiple physical activities in the physical space that are dependent on one another (see figure 7) to ensure that these activities are cascaded in the digital space as well [24]. *Applicability*: complex environments and systems with causal effects between events, such as supply chain management, where an event at one stage of the process may have cascading effects. *Advantages and Limitations*: It provides integration and alignment of interdependent processes but may require intricate mapping and comprehension of events and their causal effects, which can be challenging.

 ISSN 2723-6471

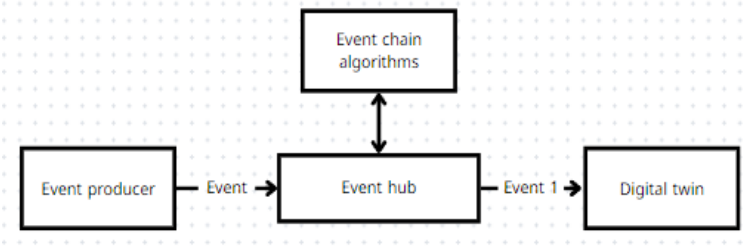

Figure 7. Event-Chain Pattern

3.7. Data-Driven Synchronization Pattern

Overview: synchronization is triggered based on data characteristics sent/received by the physical object. *Mechanics*: data and its characteristics, such as volume, frequency, and criticality, which are received or sent by the physical object, are analyzed to trigger synchronization (see figure 8). This pattern adjusts synchronization frequency based on predefined data requirements and criteria [9], [15], [20], [25]. *Applications*: Applicable in data-intensive systems where the nature of data affects synchronization, such as big data and large-scale IoT systems. *Advantages and Limitations*: Based on the analysis of data characteristics, it can potentially reduce synchronization overhead. However, dynamically categorizing data and responding to various data types and volumes can be challenging.

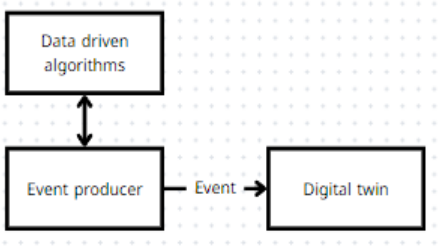

Figure 8. Data-Driven Pattern

3.8. Context-Aware Synchronization Pattern

Overview: Synchronization is based on the physical asset's current context and/or surrounding environmental factors. As can be seen in figure 9, a context-aware agent is deployed to monitor the system's context and trigger adaptation when contextual changes occur. *Mechanics*: Appropriate monitoring mechanisms (such as sensors) at the physical object detect relevant contextual/environmental changes and then trigger the synchronization process [20], [26]. This pattern can be implemented by an autonomic manager to monitor the current context and trigger synchronization based on environmental changes [16]. *Applicability*: This concept is applicable in domains where external context/environment impacts the operation of the physical asset, such as adaptive building management systems that respond to occupancy and environmental changes. *Advantages and Limitations*: This approach considers external/surrounding factors related to the physical object. However, it can be challenging to precisely capture and handle many contextual and environmental factors.

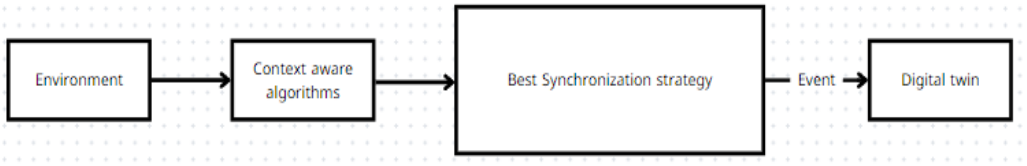

Figure 9. Context-Aware Pattern

3.9. Multi-Modal Synchronization Pattern

Overview: As shown in figure 10, this pattern integrates and synchronizes data from multiple sources (such as mobile edge nodes), enhancing the accuracy and richness of the digital twin's representation. *Mechanics*: Data from multiple sources in the physical space are collected and integrated at an appropriate point to update the digital space [2], [3]. *Applicability*: complex systems with multiple and diverse data sources, such as IoT applications and Edge computing.

 ISSN 2723-6471

Advantages and Limitations: Data collection and integration in the physical space reduce data volume in the digital space. However, the integration of possible heterogeneous data types and sources increases complexity.

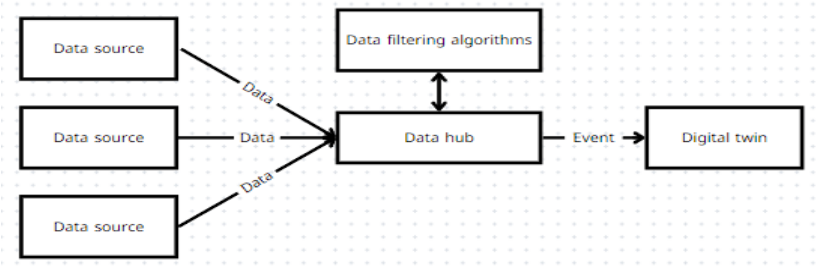

Figure 10. Multi-Modal Pattern

3.10. Asynchronous Synchronization Pattern

Overview: This pattern provides non-simultaneous synchronization, so updates to the digital twin are not immediate. As shown in figure 11, the digital twin may receive multiple non-immediate synchronization events. *Mechanics*: This presents an opportunity to obtain updated data, no matter when required, and for unsynchronized processes that are not at the same point in time. In their common configuration scheme, it is possible to optimize the use of hardware resources and take advantage of scheduling updates during off-peak times. *Applicability*: Applicable in systems where immediate synchronization is not required and delays in updates are acceptable. *Advantages and Limitations*: This pattern can help optimize network and computational resources by scheduling updates during off-peak times. However, it may temporarily cause the digital twin to be out of synchronization with the physical object.

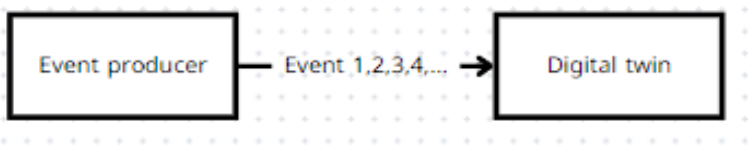

Figure 11. Asynchronous Pattern

3.11. Manual Synchronization Pattern

Overview: synchronization is triggered through manual human intervention for explicit control over digital twin updates. *Mechanics*: This pattern relies on human intervention for synchronization, in which an operator (see figure 12) manually initiates and oversees the synchronization process (e.g., via a Graphic User Interface). *Applicability*: This pattern can be used in environments that are not fully automated, where automatic synchronization is difficult to achieve and manual checks or decisions are needed. It can also be incorporated into the system design as an alternative pattern in which synchronizations by other patterns fail. *Advantages and Limitations*: Incorporates human expertise but is labor-intensive and prone to human errors.

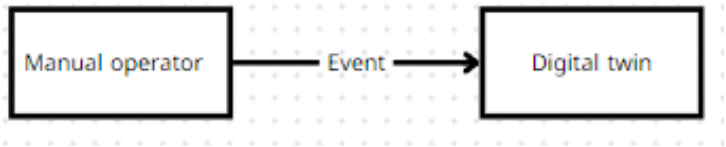

Figure 12. Manual Pattern

4. Result and Discussion

The synchronization patterns for digital twins were validated through simulations of the Automated Guided Vehicle, Emergency Monitoring, and Distributed Factory Automation architectures [27], [28]. The simulations were implemented using Sparx Enterprise Architecture. A summary of these architectures is provided as follows. The factory contains an assembly line that consists of a sequence of several workstations for parts assembly. Each workstation receives parts from its predecessor, processes each part, and then sends the processed part to its successor workstation for further processing. Therefore, a part is processed by several workstations in sequence. Automated Guided Vehicles

 ISSN 2723-6471

(AGVs) move along a track to load/unload parts to/from workstations. The emergency monitoring system monitors the environment in the factory via various sensors and generates alarms for undesirable situations (e.g., fire). An alarm service stores these alarms so that they can be viewed by operators. The following sections describe various simulations of the patterns that synchronize the digital twins corresponding to physical objects in these architectures.

4.1. Synchronizing the State of AGVs Digital Twins

In our simulations, each AGV is associated with a digital twin. The state of the digital twin for each AGV is synchronized using the Hybrid Synchronization Pattern. Figure 13 shows the augmented static model for this system that incorporates AGV digital twins. To synchronize the state of an AGV, we augmented the design with a wrapper (VehicleWrapper in figure 13) that intercepts all messages to/from the vehicle. This wrapper class is embedded in the connector of the AGV component, which tracks its current state [29] and triggers synchronization as follows. The wrapper forwards the messages to their intended destination for actual processing but also uses these messages to track the current state of the AGV using a state machine (see figure 14). Utilizing this state machine, the wrapper accurately identifies the distinct states at which the digital twin should be synchronized. Note that at each state transition, the action is to forward the received message and synchronize the digital twin. Therefore, this is an example of event-driven synchronization. In addition, the vehicle may remain in the Idle state indefinitely if there are no tasks from the supervisory system. In this case, and as shown in figure 14, the Idle state executes an activity to synchronize the state of the digital twin every N second, an example of Time-Driven synchronization.

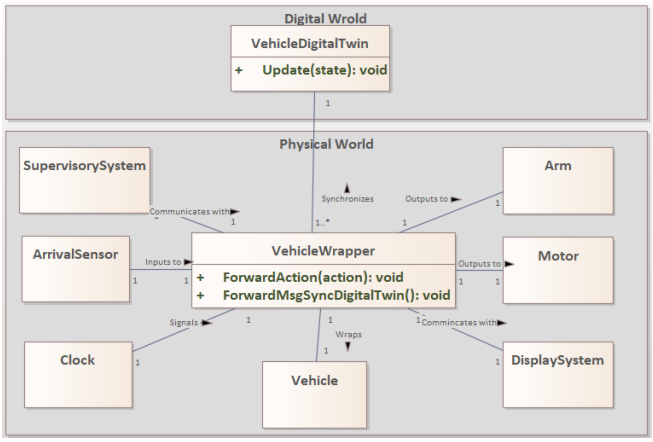

Figure 13. A static view of the augmented AGV system.

While conducting the simulation, trace logs are collected and analyzed to ensure proper synchronization of the digital twin. Analysis of the trace log indicates proper synchronization triggered by VehicleWrapper such that with each important arriving message it receives (which expects to trigger a change in the physical vehicle), the wrapper triggers the synchronization process and forwards the message to the physical vehicle for further processing.

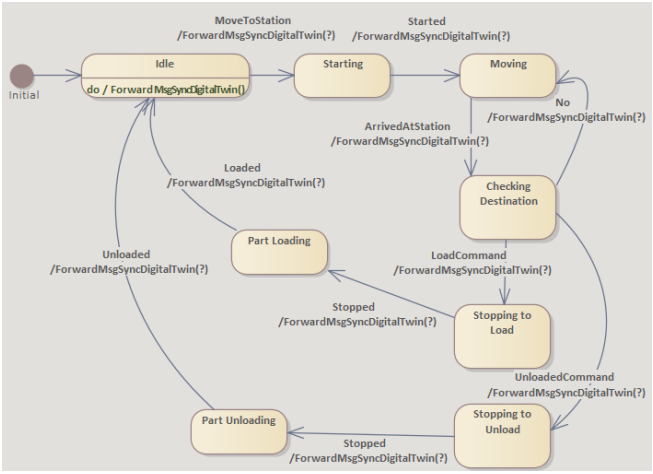

Figure 14. the state machine embedded in VehicleWrapper for state tracking and synchronization.

 ISSN 2723-6471

4.2. Synchronizing the State of Workstation Digital Twins

In our simulations, each line workstation in the factory is associated with a digital twin. Since each workstation receives parts from its predecessor, processes them, and then sends them to its successor for further processing, we applied the Event-Chain synchronization pattern, as shown in figure 15, as this is considered a sequential interaction between the various workstations. In this figure, the sequential workstations are associated with a wrapper that monitors incoming events from/to these workstations. The wrapper intercepts these messages and updates the state of the digital twin at these points.

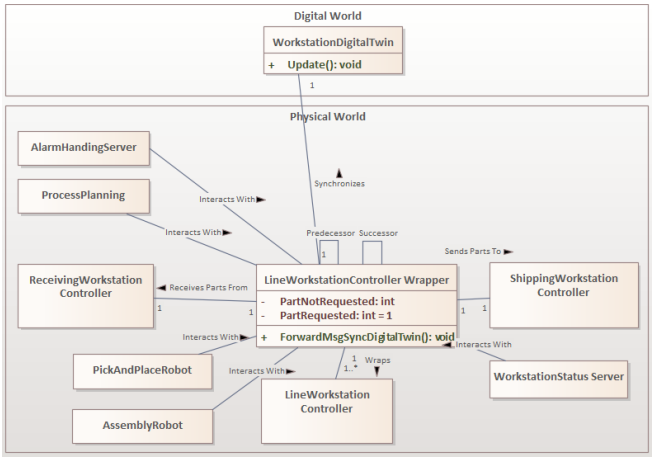

Figure 15. A fragment of the static view for the Factory Automation system.

Analysis of the collected simulation log indicates that the WorkstationControllerWrapper intercepted incoming messages that affect the state of the physical workstation (e.g., part arriving, robot picking, and assembling part events). As a result, it triggered the synchronization process with the digital twin and forwarded these events to the workstation controller. We incorporated the Context-Aware synchronization pattern into our simulation so that when an alarm is generated by the emergency monitoring system, the synchronization frequency is increased between physical assets in the factory and their digital twins. When alarms are cleared, normal synchronization is resumed.

5. Conclusion and Future Work

This paper presents a catalog of 11 synchronization patterns for digital twin systems. The patterns in this catalog are identified based on the analysis of the results obtained from the literature review. We provide an overview of each pattern, its applicability, and advantages and limitations, where this research aims to fill the gap identified by prior research works. Furthermore, we evaluated these patterns through simulations of multiple software architectures. In our simulator, we consider embedding the various patterns in component connectors rather than actual application components to minimize changes in these components.

As part of our future work, we are interested in investigating how the quality attributes of these synchronization patterns (such as performance, efficiency, and scalability) can be quantitatively evaluated through experimental analysis. Furthermore, we plan to investigate how autonomic feedback loops can be incorporated to integrate multiple synchronization patterns to cope with the more complicated system needs more efficiently, where the goal is to design an autonomic manager capable of choosing the most appropriate set of synchronization patterns that are best suited for the system. Finally, we are interested in exploring how Machine Learning enhancements can be introduced to provide significant improvements in these patterns synchronizing, particularly regarding those that pertain to predictive analytics, real-time decision support, and adaptive learning for system optimization.

6. Declaration

6.1. Author Contributions

Conceptualization: W.A. and E.A.; Methodology: W.A. and E.A.; Software: W.A.; Validation: W.A. and E.A.; Formal Analysis: W.A.; Investigation: W.A.; Resources: E.A.; Data Curation: W.A. and E.A.; Writing Original Draft

Preparation: W.A.; Writing Review and Editing: W.A. and E.A.; Visualization: W.A.; All authors have read and agreed to the published version of the manuscript.

6.2. Data Availability Statement

The data presented in this study are available on request from the corresponding author.

6.3. Funding

The authors received no financial support for the research, authorship, and/or publication of this article.

6.4. Institutional Review Board Statement

Not applicable.

6.5. Informed Consent Statement

Not applicable.

6.6. Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

[1] H. Zipper, “Real-Time-Capable Synchronization of Digital Twins,” *IFAC-Pap.*, vol. 54, no. 4, pp. 147–152, Jan. 2021, doi: 10.1016/j.ifacol.2021.10.025.

[2] T. Zhou, X. Zhang, B. Kang, and M. Chen, “Multimodal fusion recognition for digital twin,” *Digit. Commun. Netw.*, vol. 10, no. 2, pp. 337–346, Apr. 2024, doi: 10.1016/j.dcan.2022.10.009.

[3] W. Liu et al., “A 5M Synchronization Mechanism for Digital Twin Shop-Floor,” *Chin. J. Mech. Eng.*, vol. 36, no. 1, Art. no. 136, pp. 1–29, Nov. 2023, doi: 10.1186/s10033-023-00965-8.

[4] M. Cunha, R. Mendes, and J. P. Vilela, “A survey of privacy-preserving mechanisms for heterogeneous data types,” *Comput. Sci. Rev.*, vol. 41, Art. no. 100403, pp. 1–15, Aug. 2021, doi: 10.1016/j.cosrev.2021.100403.

[5] A. Abouelrous, L. Bliek, and Y. Zhang, “Digital twin applications in urban logistics: an overview,” *Urban Plan. Transp. Res.*, vol. 11, no. 1, Art. no. 2216768, pp. 1–26, Jun. 2023, doi: 10.1080/21650020.2023.2216768.

[6] A. Fuller, Z. Fan, C. Day, and C. Barlow, “Digital Twin: Enabling Technologies, Challenges and Open Research,” *IEEE Access*, vol. 8, pp. 108952–108971, May 2020, doi: 10.1109/ACCESS.2020.2998358.

[7] P. Jia, X. Wang, and X. Shen, “Digital-Twin-Enabled Intelligent Distributed Clock Synchronization in Industrial IoT Systems,” *IEEE Internet Things J.*, vol. 8, no. 6, pp. 4548–4559, Mar. 2021, doi: 10.1109/JIOT.2020.3029131.

[8] M. Attaran, S. Attaran, and B. G. Celik, “The impact of digital twins on the evolution of intelligent manufacturing and Industry 4.0,” *Adv. Comput. Intell.*, vol. 3, no. 3, Art. no. 11, pp. 1–15, Jun. 2023, doi: 10.1007/s43674-023-00058-y.

[9] S. T. Johansen, P. Unal, Ö. Albayrak, E. Ikonen, K. J. Linnestad, S. Jawahery, A. K. Srivastava, and B. T. Løvfall, “Hybrid and cognitive digital twins for the process industry,” *Open Eng.*, vol. 13, no. 1, Art. no. 20220418, pp. 1–13, Mar. 2023, doi: 10.1515/eng-2022-0418.

[10] C. E. B. López, “Real-time event-based platform for the development of digital twin applications,” *Int. J. Adv. Manuf. Technol.*, vol. 116, no. 3, pp. 835–845, Jun. 2021, doi: 10.1007/s00170-021-07490-9.

[11] F. Akbarian, E. Fitzgerald, and M. Kihl, "Synchronization in digital twins for industrial control systems," *in Proc. 16th Swedish National Computer Networking Workshop (SNCNW 2020)*, vol. 2020, no. May, pp. 1–4, 2020, doi: 10.48550/arXiv.2006.03447.

[12] J. Yu, A. Alhilal, T. Zhou, P. Hui and D. H. K. Tsang, "Attention-based QoE-aware Digital Twin Empowered Edge Computing for Immersive Virtual Reality," *IEEE Transactions on Wireless Communications*, vol. 2024, no. Mar., pp. 1–16, 2024, doi: 10.1109/TWC.2024.3380820.

[13] S. Shang, Z. Yu, K. Jiao, Y. Huang, H. Guo, and G. Wang, “Design of Cross-Platform Information Retrieval System of

Library Based on Digital Twins," *Comput. Intell. Neurosci.*, vol. 2022, no. 1, Art. no. 7999091, pp. 1–10, Sep. 2022, doi: 10.1155/2022/7999091.

[14] B. Tan and A. Matta, "The digital twin synchronization problem: Framework, formulations, and analysis," *IISE Trans.*, vol. 56, no. 6, pp. 652–665, Jun. 2024, doi: 10.1080/24725854.2023.2253869.

[15] G. Sahar, K. A. Bakar, S. Rahim, N. A. K. K. Khani, and T. Bibi, "Recent Advancement of Data-Driven Models in Wireless Sensor Networks: A Survey," *Technologies*, vol. 9, no. 4, Art. No. 76, pp. 1–26, Oct. 2021, doi: 10.3390/technologies9040076.

[16] J. O. Kephart and D. M. Chess, "The vision of autonomic computing," *Computer*, vol. 36, no. 1, pp. 41–50, Jan. 2003, doi: 10.1109/MC.2003.1160055.

[17] A. Phua, C. H. J. Davies, and G. W. Delaney, "A digital twin hierarchy for metal additive manufacturing," *Comput. Ind.*, vol. 140, Art. no. 103667, pp. 1–14, Sep. 2022, doi: 10.1016/j.compind.2022.103667.

[18] Y. Han et al., "A Dynamic Hierarchical Framework for IoT-Assisted Digital Twin Synchronization in the Metaverse," *IEEE Internet Things J.*, vol. 10, no. 1, pp. 268–284, Aug. 2023, doi: 10.1109/JIOT.2022.3201082.

[19] B. Biller and S. Biller, "Implementing Digital Twins That Learn: AI and Simulation Are at the Core," *Machines*, vol. 11, no. 4, Art. no. 425, pp. 1–18, Mar. 2023, doi: 10.3390/machines11040425.

[20] K. Hribernik, G. Cabri, F. Mandreoli, and G. Mentzas, "Autonomous, context-aware, adaptive Digital Twins—State of the art and roadmap," *Comput. Ind.*, vol. 133, Art. no. 103508, pp. 103508–103508, Dec. 2021, doi: 10.1016/j.compind.2021.103508.

[21] E. Albassam and J. Porter, "A Decentralized Self-Optimization Approach for Distributed Component-Based Software Systems," *in Proceedings of the 2021 10th International Conference on Software and Computer Applications, in ICSCA '21. New York, NY, USA: Association for Computing Machinery*, vol. 2021, no. Jul., pp. 124–130, 2021. doi: 10.1145/3457784.3457802.

[22] D. Garlan, S.-W. Cheng, A.-C. Huang, B. Schmerl, and P. Steenkiste, "Rainbow: architecture-based self-adaptation with reusable infrastructure," *Computer*, vol. 37, no. 10, pp. 46–54, Oct. 2004, doi: 10.1109/MC.2004.175.

[23] K. Bakliwal, M. H. Dhada, A. S. Palau, A. K. Parlikad, and B. K. Lad, "A Multi Agent System architecture to implement Collaborative Learning for social industrial assets," *IFAC-Pap.*, vol. 51, no. 11, pp. 1237–1242, 2018, doi: 10.1016/j.ifacol.2018.08.421.

[24] A. Ricci, A. Croatti, S. Mariani, S. Montagna, and M. Picone, "Web of Digital Twins," *ACM Trans Internet Technol*, vol. 22, no. 4, Art. no. 101, pp. 1–30, Nov. 2022, doi: 10.1145/3507909.

[25] J. Friederich, D. P. Francis, S. Lazarova-Molnar, and N. Mohamed, "A framework for data-driven digital twins of smart manufacturing systems," *Comput. Ind.*, vol. 136, Art. no.103586, pp. 1–13, Apr. 2022, doi: 10.1016/j.compind.2021.103586.

[26] M. Rico, M. L. Taverna, M. R. Galli, and M. L. Caliusco, "Context-aware representation of digital twins' data: The ontology network role," *Comput. Ind.*, vol. 146, Art. no.103856, pp. 1–14, Apr. 2023, doi: 10.1016/j.compind.2023.103856.

[27] H. Gomaa, Software Modeling and Design: UML, Use Cases, Patterns, and Software Architectures. Cambridge: Cambridge University Press, 2011. doi: 10.1017/CBO9780511779183.

[28] H. Gomaa, Designing concurrent, distributed, and real-time applications with UML. Boston: Addison-Wesley, 2000.

[29] E. Albassam, H. Gomaa, and D. A. Menascé, "Variable Recovery and Adaptation Connectors for Dynamic Software Product Lines," *in Proceedings of the 21st International Systems and Software Product Line Conference - Volume B, in SPLC '17. New York, NY, USA: Association for Computing Machinery*, vol. 1, no. Sept., pp. 123–128, 2017, doi: 10.1145/3109729.3109742.

[30] H. Wu, P. Ji, H. Ma, and L. Xing, "A Comprehensive Review of Digital Twin from the Perspective of Total Process: Data, Models, Networks and Applications," *Sensors*, vol. 23, no. 19, Art. no.8306, pp. 1–20, Jan. 2023, doi: 10.3390/s23198306.

[31] D. Jones, C. Snider, A. Nassehi, J. Yon, and B. Hicks, "Characterising the Digital Twin: A systematic literature review," *CIRP J. Manuf. Sci. Technol.*, vol. 29, no. part A, pp. 36–52, May. 2020, doi: 10.1016/j.cirpj.2020.02.002.

[32] Z. Shen, F. Arraño-Vargas, and G. Konstantinou, "Artificial intelligence and digital twins in power systems: Trends, synergies and opportunities," *Digit. Twin*, vol. 2, no. 11, pp. 1–29, Mar. 2022, doi: 10.12688/digitaltwin.17632.2.