



An Agent Design Pattern Catalogue for the Entanglement of Digital Twins

Elena Pretel¹ , Elena Navarro¹ , Víctor López-Jaquero¹ ,
José J. González-García¹ , and Schahram Dustdar² 

¹ LoUISE Research Group, I3A, University of Castilla-La Mancha, 02071 Albacete, Spain
{mariaelena.pretel, elena.navarro, victormanuel.lopez,
josejesus.gonzalez}@uclm.es

² TU Wien, 1040, Vienna, Austria, and UPF ICREA, 08018 Barcelona, Spain
dustdar@dsg.tuwien.ac.at

1 Introduction

A Digital Twin (DT) is defined as a digital representation of a physical entity that allow real-time monitoring, simulation, prediction and control of the behaviour of the modelled entities [23]. This concept, originally coined by Grieves and Vicker in 2003 [11], has gained significant attention in recent years due to its potential applications across various domains, including industrial production, healthcare, smart cities, and energy management [12]. For instance, a DT of a wind turbine can continuously monitor operating parameters such as wind speed, rotor speed, etc. By analysing these data, the DT can detect anomalies and automatically activate some control mechanisms to stop the physical turbine remotely, thus preventing malfunctioning [27].

As indicated by Minerva et al. [23], the essential property of DTs that allows the physical and cybernetic convergence is called *entanglement*. This property defines the relationship between the physical twin and its digital replica (DT). This relationship implies a constant and as far as possible, real-time flow of information, whereby the DT accurately reflects the state of the physical twin. This property is recognized as one of the most important aspects in several standardization approaches, including ISO 23247 [15] that provides a general framework for the implementation of DT in industrial manufacturing environments highlighting the importance of maintaining efficient synchronization between physical and digital entities for effective system control and monitoring. The entanglement property is based on three aspects [23]:

- *Connectivity* (C): there should be a direct or indirect means of communicating state changes and data between the physical and digital twin. Transmission can be direct if the physical twin can communicate directly with its DT or indirect if it depends on a third entity to send the information.
- *Association* (A): it describes the direction of data exchange between the physical twin and its DT. This association can be unidirectional, for example, sensors that only transmit data to the DT, or bidirectional, the DT can also act on the physical twin through actuators in order to modify its behaviour.

- *Promptness (P)*: data updates must occur at times compatible with the intended use of the DT. In critical applications, such as medical or industrial, continuous data flow is required, however, in other cases, periodical or event-driven updates may be sufficient.

Based on these three aspects, the degree of entanglement can vary [23]:

- *Weak entanglement* means that data from the physical twin are inferred indirectly by analysing data from the surrounding environment (C). For example, through observation of the environment or analysis of external sources, such as social networks.
- *Simple entanglement* means that there is a unidirectional (A) or no continuous link between the physical twin and its DT (P), allowing partial synchronization without instant feedback.
- *Strong entanglement* means that the physical twin is continuously linked to the DT (P) and moreover the link is bidirectional (A), so, the DT has also the ability to modify or update the status of the physical twin.

Recently, the use of Multi-Agent Systems (MAS) has emerged as a promising architectural approach to address the development of DTs [26]. MAS are made up of multiple agents that interact with each other and with their environment to achieve goals [4]. The properties of MAS can support the development of DT properties [26]. For instance, the *data transmission frequency* MAS property [4] can help to the development of the entanglement DT property [26]. This enables the development of DTs that not only reflect the state of their physical twin, but can also reason, learn, and interact with others to achieve complex goals.

Although the use of MAS has become common in the design and implementation of DTs [26], a gap still exists on how agents can be designed to adapt to different levels of entanglement. There are no proposals that provide a catalogue of agent design patterns related to the entanglement aspects of DTs. The aim of this paper is to systematically investigate the design of agents, focusing on how they support and manage the entanglement between the physical and digital twin. Therefore, the research questions that this paper addresses are the following:

- **RQ1**: Which agent design patterns can be used to support the development of the entanglement property of DT?
- **RQ2**: How can software architects be guided in the selection of design patterns to achieve different degrees of entanglement of the DT?

This article contains three main contributions. First, we propose a catalogue of design patterns based on MAS patterns found in the literature that guide the development of the entanglement DT property. Second, we propose a decision model that guides architects toward which agents they need to include in their architecture to support the DT entanglement expected. Finally, we evaluate the level of detail and completeness of the catalogue of design patterns by implementing a case study demonstrating that our pattern catalogue offers valuable insights for agent-based DT development.

The rest of this paper is structured as follows: Sect. 2 compares this work with the related work. Then, Sect. 3 explains the research methods applied in this study. Next,

Sect. 4 describes the catalogue of agent design patterns and Sect. 5 discusses the decision model for pattern selection based on the DT entanglement aspects. Then, Sect. 6 shows the practical application of the catalogue of design patterns proposed, the discussion and the threats to the validity of our study. Finally, Sect. 7 summarises the conclusions and future work.

2 Related Work

Design patterns are a crucial part of software engineering, especially in the context of system architecture and the development of complex systems. Design patterns are typical solutions to common problems in software design [31]. With the increasing use of MAS for the development of software systems, various works have focused on the development of design patterns for agent-based systems. For example, Oluyomi et al. [25] highlight the different levels of complexity and abstraction that are introduced in MAS and the differences in the way agents are implemented. Marks et al. [21] survey some agent design patterns for assistive systems in different domains. Luder et al. [20] introduces design patterns for the control of agent-based production systems. Liu et al. [18] propose a collection of patterns for agents-based on foundational models

In the field of DTs, few works have proposed specific design patterns. For example, Tekinerdogan et al. [30] propose a set of digital twin architecture design patterns. Then, Sulaiman Khail et al. [29] proposes a set of design patterns that enable transitions from DT with weak entanglement till strong entanglement. Although not exclusively focused on design patterns, Minerva et al. [23] provide a comprehensive review of the technical characteristics and architectures of DTs, indicating common patterns for the implementation of DTs in the context of IoT.

Regarding design patterns for agents in the context of DT, a comprehensive review of the literature reveals that, despite the increasing development of DT using MAS, no prior work has systematically formalized a catalogue of agent design patterns explicitly oriented toward supporting the entanglement property. This research aims to fill this gap by proposing a catalogue of agent design patterns, detailed in Sect. 4, specifically developed to address the different levels of entanglement: weak, simple, and strong.

3 Methodology

This section discusses the research method followed in this study. Figure. 1 shows the research methodology which combines two research methods: Systematic Literature Review (SLR) and Grounded Theory (GT). By integrating these two methods, recurrent structures and agent behaviours related to the entanglement property of DTs were explored.

First, this paper uses the Systematic Literature Review (SLR) on DT and MAS conducted by Pretel et al. [26]. This SLR follows the guidelines proposed by Kitchenham et al. [17]. therefore, the authors focused on the available primary research papers and selected relevant papers based on a set of inclusion and exclusion criteria. Moreover, they conducted a quality assessment of the selected materials to ensure the quality of the

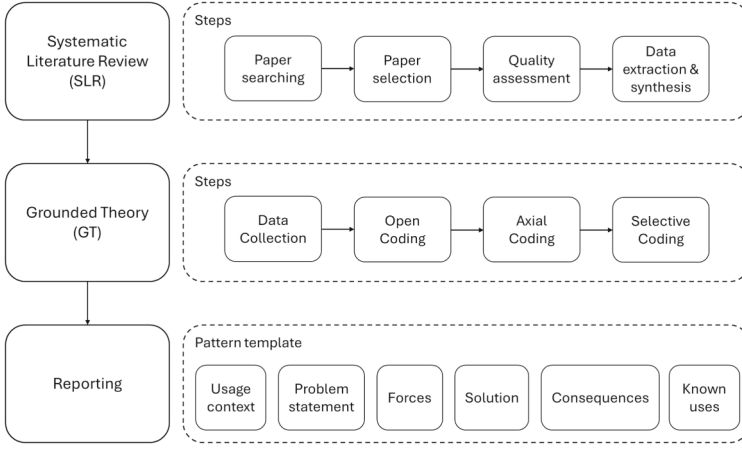


Fig. 1. Methodology (adapted from [18])

papers. Finally, in this SLR, 64 studies were included for data extraction and synthesis. This study has used the information extracted from these papers.

Then, in order to further enrich the information gathered to define patterns, in this study we extended our research approach by including a Grounded Theory (GT) [8]. This combination allowed us to complement the SLR results with additional findings derived from a process of inductive and iterative analysis. Firstly, we conducted additional searches focusing on MAS design pattern works that were not captured in the SLR [26]. Next, open coding was applied to the collected works to identify design patterns related to the entanglement property, followed by axial coding to establish connections between the patterns and the three aspects of entanglement (connectivity, promptness and association). Finally, selective coding was applied to create a catalogue of design patterns related to the entanglement property of the DT.

The reporting of the patterns has been carried out using the extended template of Meszaros [22]. It includes the name of the pattern, a brief summary, the context of use, a problem statement, the forces, the solution, its consequences, the related design patterns, and some examples of known uses of the pattern in the real world.

This research methodology combining SLR and GT provided a richer understanding of the design principles and architectural patterns used to develop the entanglement of the DT. By combining the results of both research methods, we ensured that the identified patterns are based on empirical findings and, at the same time, are theoretically grounded. This analysis provides the basis for the next section of this article, in which the derived design patterns are presented.

4 Agent Design Pattern Catalogue for the Entanglement

To address the **RQ1**, this Section presents a catalogue of agent design patterns aimed at supporting the development of the different types of entanglement in DTs: weak, simple, and strong. Each pattern is systematically described by using the extended Meszaros format [22].

4.1 Indirect Observation Pattern for Weak Entanglement

Summary: This pattern implements an agent that infers the state of a physical twin indirectly by analysing the data from its surrounding environment or external tools and periodically sends information to the agents of the DT to update its state. This pattern is characteristic of *weak entanglement*.

Context: In some cases, physical twins lack integrated sensors or cannot establish a direct communication channel with their DT. However, their state can be inferred by observing their surrounding environment. For example, the state of physical twin can be monitored by a camera, extracting some relevant information and transmitting it to the DT.

Problem: How to synchronize the DT when there is no direct connection with its physical twin?

Forces

- Sensor limitation: The physical twin has no embedded sensors or direct connectivity to its DT.
- Data inference: The agent into the physical twin needs to infer its status from environmental data.
- Synchrony requirement: The DT must be updated without direct communication.

Solution: This problem can be solved by implementing an architecture similar to the *MAS Observer Pattern* [24]. First, it is necessary implementing a *subject agent* that observes the environment or analyses some external data sources related to the physical twin. This agent processes these data and transmits the information to the *observer agents*, which are in the DT. As the *subject agent* has to infer the state of the physical twin, it will transmit data periodically rather than in real time or when events occur. Finally, when the *observer agents* receive the information, they update the status of the DT.

When multiple heterogeneous external sources are available, such as cameras, social networks, public APIs, etc., the *subject agent* is implemented following the *MAS Aggregation Pattern* [6] to collect, integrate and harmonise the data before sending it to the DT. This improves the accuracy of the inferred state of the physical twin.

Consequences

Benefits:

- No sensor dependency: Allows operation without intrusive or costly sensors.

Drawbacks:

- Inaccuracy: DT reflects a less accurate representation of the physical twin than using direct measurement.
- Risk of error: Agents may make mistakes by inferring incorrect states of the physical twin.

Known Uses

- Smart city: DT for monitoring the population movement using cameras [32].

4.2 Continuous Direct Data Dispatcher Pattern for Simple and Strong Entanglement

Summary: This pattern implements an internal agent into the physical twin that continuously and directly monitors and transmits the state of the physical twin and any relevant data to the DT through appropriate networks and communication protocols. This pattern is characteristic of *strong entanglement* supporting the direct continuous link from the physical twin to the DT. In addition, this pattern is also characteristic of *simple entanglement* of DTs whenever continuous communication from the physical twin to the DT is required.

Context: Sometimes, the physical twin has embedded computational and networking capabilities to communicate continuously and directly its information to its DT. This occurs in smart devices with sensors that report their status without relying on external observation.

Problem: How to synchronize the DT when the physical twin has direct and continuous communication capabilities?

Forces

- Technology availability: The physical twin has embedded sensors and network connectivity.
- Immediate synchronisation: The DT needs to continuously receive information.
- Compatibility with IoT architecture: The solution aligns with modern IoT architectures.

Solution: An agent is implemented in the physical twin that collects data and transmits it directly to the DT. This can be implemented by using the *MAS Publish-Subscribe Pattern* [1]. Here, the *publisher agent* is responsible for collecting and sending data in real time to the *message broker*. This broker is responsible for retransmitting messages to the *subscriber agents*. Each agent is subscribed to relevant topics, thus receiving only useful information. This architecture with an intermediary allows for a reduction in message traffic, which is essential for efficient continuous communication.

Consequences

Benefits:

- Fidelity: The DT reflects the state of the physical twin with high accuracy.
- Reliability: The agent into the physical twin transmits data directly and constantly to the DT.

Drawbacks:

- Dependency: The DT depends on a reliable network to receive data correctly.

Related Design Patterns: It should be combined with the *Action Dispatcher Pattern* to enable a DT with *strong entanglement*.

Known Uses

- Manufacturing: Industrial machinery with embedded sensors that update their DT [10].
- Health: Medical devices that send data to the DT of the patient [13].

4.3 Event-Driven Direct Data Update Pattern for Simple Entanglement

Summary: This pattern implements an agent in the physical twin that responds to specific events to directly send information to agents in the DT to update its state. This pattern is characteristic of *simple entanglement* of DTs that require event-driven communication from the physical twin to the DT.

Context: In some cases, it is not necessary to continuously send data from the physical twin, but it is sufficient to update the state of the DT whenever a relevant event occurs in the physical twin.

Problem: How to ensure that the DT is updated whenever relevant events occur in the physical twin?

Forces

- Transmission efficiency: The physical twin agent only transmits data when a significant event occurs.
- Critical detection: The physical twin agent must detect events accurately.

Solution: The architecture of the *MAS Event Dispatcher pattern* [3] is used to provide a solution to this problem. This pattern defines explicit agreements between agents on which events should be notified and how they should be handled. So, an *event-driven reactive agent* is implemented in the physical twin. This agent continuously monitors the state of the physical twin and detects predefined events. When an event occurs, the *event-driven reactive agent* uses the architecture of *MAS Observer pattern* [24] to notify the DT by sending the new information to the *observer agents* of the DT. Thus, the *event-driven reactive agent* acts as a *subject agent*. Finally, when the *observer agents* receive the information, they update the status of the DT.

The use of the *MAS Event Dispatcher pattern* [3] allows the *event-driven reactive agent* to dynamically decide whether to send the information to the DT or not, depending on the criticality of the event and the availability of resources.

Consequences

Benefits:

- Low latency: The DT is immediately updated in case of critical events.
- Effectiveness: The agent into the physical twin minimises the use of resources by sending data only when necessary.

Drawbacks:

- Complex design: The design of the agent into the physical twin requires defining relevant events precisely.
- Sensitivity: The DT can lose information if an event is not detected correctly.

Known Uses

- Farming: data collected from plants are sent to the DT once a day or whenever certain events occur [28].

4.4 Periodic Direct Data Sending Pattern for Simple Entanglement

Summary: This pattern implements an agent in the physical twin that sends periodic status updates to the DT. It uses a model based on fixed time intervals to ensure that the DT receives information on a regular basis. This pattern is characteristic of *simple entanglement* of DTs that require periodic communication from the physical twin to the DT.

Context: In some cases, the physical twin does not experience abrupt or critical changes, but it is important to keep constant synchronisation with the DT. Therefore, a periodic communication mechanism is necessary.

Problem: How to ensure that the DT receives regular updates from the physical twin without overloading the network?

Forces

- Planning: The agent into the physical twin can schedule data dispatches at fixed intervals.
- Latency tolerance: The DT can accept delays between updates.

Solution: The architecture of the *MAS Observer Pattern* [24] is used to provide a solution to this problem. First, it is necessary implementing a *subject agent* that periodically collects the data from the physical twin and sends this information to the *observer agents* at predefined time intervals. In the DT, these *observer agents* update the status of the DT with the new data received.

Consequences

Benefits:

- Consistency: The DT receives regular updates.

- Efficiency: The agent into the physical twin uses less network resources than a continuous transmission.

Drawbacks:

- Dephasing: The DT can become out of date between transmissions.
- Potential inefficiency: The DT could waste resources if the intervals are too short, and significant changes occur less often.

Known Uses

- Health: Medical devices that alert to changes in the health status of the patient [2].

4.5 Action Dispatcher Pattern for Simple and Strong Entanglement

Summary: This pattern allows agents in the DT to send commands to the physical twin, allowing its behaviour or state to be modified by actuators. This pattern is characteristic of *strong entanglement* supporting the link from the DT to the physical twin. In addition, this pattern is also characteristic of *simple entanglement* of DTs that require communication from the DT to the physical twin.

Context: Occasionally, the DT not only needs to reflect the state of the physical twin, but also to interact with it, emitting decisions that translate into physical actions, such as activating an alarm, adjusting a parameter or turning on a device.

Problem: How to enable the DT to communicate instructions to its physical twin to produce real changes in the physical world?

Forces

- Actuation capability: The DT can send commands to modify the behaviour of the physical twin.

Solution: This pattern can be implemented by using several MAS design patterns together. First, an agent implemented in the DT is required following the *MAS Command Dispatcher Pattern* [5], which encapsulates the action as a message with some defined parameters. Then, to send messages from the DT and to its physical twin, the *MAS Publish-Subscribe* [1] can be used. Thus, an agent of the DT (*publisher agent*) sends the generated message to an intermediary (*message broker*). Finally, in the physical twin, for each actuator, an agent (*subscriber agent*) is implemented as a message listener that translates messages into actions for their execution.

Consequences

Benefits:

- Active control: The DT can correct or improve the behaviour of the physical twin.

Drawbacks:

- Complexity: The DT requires a more advanced architecture.
- Safety risk: The use of remote control could introduce security risks or errors.

Related Design Patterns: It should be combined with the *Continuous Direct Data Dispatcher* to enable a DT with *strong entanglement*.

Known Uses

- Energy: Smart homes that optimise their energy consumption according to conditions detected by the DT [14].
- Manufacturing: Robots controlled in real time by a DT [7].

5 Decision Model for Selection of Desing Patterns

To address the **RQ2**, in this Section, we propose a decision model for selecting the identified design patterns according to the three foundational aspects of entanglement: connectivity, promptness and association. These aspects allow to systematically classify the degree of entanglement into weak, simple, or strong. Fig. 2 shows this decision model.

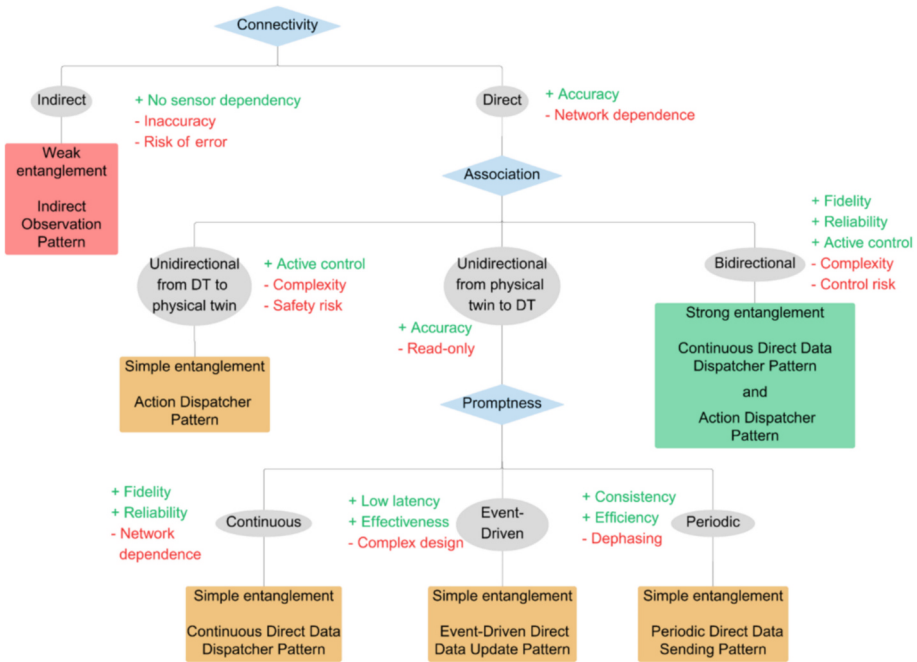


Fig. 2. Decision Model

5.1 Step 1: Determine the Type of Connectivity (C)

The first step evaluates the nature of the connection between the physical twin and the DT. Two possible outcomes are considered:

- **Indirect:** The communication between the physical twin and its DT is mediated by intermediaries. In this case, the entanglement is classified as *Weak Entanglement* due to the limited and indirect interaction and the recommended pattern is *Indirect Observation* (Sect. 4.1).
- **Direct:** The physical and DT can communicate directly without intermediaries. In this case, the decision process continues to the second step to determine the requirements of association for the DT.

5.2 Step 2: Determine the Direction of Association (A)

The second step examines the directionality of communication between the physical twin and the DT. Three possible scenarios are considered:

- **Bidirectional:** Information flows in both directions, enabling a dynamic interaction between the physical twin and its DT. In this case, the entanglement is classified as *Strong Entanglement*, so the recommended patterns are *Continuous Direct Data Dispatcher* (Sect. 4.2) for the communication from the physical twin to the DT, and the *Action Dispatcher* (Sect. 4.5) for the communication from the DT to the physical twin.
- **Unidirectional from DT to physical twin:** Information flows exclusively from the DT to the physical twin. This scenario results in *Simple Entanglement*, as the interaction is limited to one direction. The recommended pattern in this case is the *Action Dispatcher* (Sect. 4.5).
- **Unidirectional from physical twin to DT:** Information flows exclusively from the physical twin to the DT. Similar to the previous scenario, this also results in *Simple Entanglement*. However, in this case, the decision process should continue further to the third step in order to determine which requirements of promptness DT has when communicating from the physical twin to the DT.

5.3 Step 3: Determine the Required Promptness (P)

The third criterion evaluates the timing and frequency of the interactions. Three possible outcomes are considered:

- **Continuous:** Data is sent from the physical twin to the DT instantaneously, ensuring that changes in the physical twin are immediately reflected in the DT. The recommended pattern is *Continuous Direct Data Dispatcher* (Sect. 4.2).
- **Event-Driven:** Data sending occurs in response to specific events or triggers, but not continuously. The recommended pattern is *Event-Driven Direct Data Update Pattern* (Sect. 4.3).
- **Periodic:** Data sending occurs at predefined regular intervals. The recommended pattern is *Periodic Direct Pattern* (Sect. 4.4).

6 Lessons Learned from the Pattern Catalogue

In this Section, we apply some patterns in one case study. Then, we discuss several insights we learned during the pattern collection process and working with them. Finally, we discuss some threats to validity of this study.

6.1 Pattern Application

In this section, we discuss and share our experiences while using this decision model and applying the identified patterns in a case study: management of study session of children with attention deficit hyperactivity disorder (ADHD) [9]. It is a neurodevelopmental disorder characterized by persistent patterns of inattention, hyperactivity, and impulsivity that interfere with daily functioning and academic performance [19]. Children with ADHD often struggle with maintaining focus, organizing tasks, following instructions, and controlling impulses, leading to difficulties in traditional learning environments.

To tackle it, innovative approaches that leverage advanced technologies are essential. One promising solution is the application of DT, because it can help to personalise and optimise study sessions based on individual needs. In this work, the DT is developed by following a MAS approach [26]. By leveraging real-time data, that is a continuous data flow, this DT can dynamically adjust the time of the study sessions, ensuring they align with the child's current needs and optimising focus and well-being.

Decision Model Application. Starting from the previously described DT needs, our decision model is applied:

- *Step 1: Determine the type of connectivity (C).* In this case, the physical twin has sensors that send directly data from the child and from the environment to the DT. Therefore, the connectivity is *direct*.
- *Step 2: Determine the direction of association (A).* In this case, a *bidirectional* association is required. This is characteristic of a *strong entanglement* (see Sect. 1), so *continuous* communication between the physical and the DT is essential (P). The DT must monitor the child's physical parameters and environmental variables in real time to ensure accurate and timely updates to the DT. Moreover, the DT also send commands to influence the environment and modify some session parameters. This bidirectional link ensures that the DT actively contributes to improving the child's study conditions.

Therefore, if we apply the decision model shown Fig. 2 with *direct connectivity*, *bidirectional association* and *continuous promptness*, the recommended patterns are the *Continuous Direct Data Dispatcher* (Sect. 4.2) for the communication from the physical twin to the DT, and the *Action Dispatcher* (Sect. 4.5) for the communication from the DT to the physical twin.

Implementation of Design Pattens. The implementation¹ has been carried out using the JaCaMo framework [32], which integrates Jason, CArtaGO, and Moise. CArtaGO

¹ GitHub repository of the developed example: <https://github.com/UCLM-LoUISE/PRJ2022-HDT-JaCaMo.git>

enabled flexible instantiation of design patterns by providing tools for designing shared artifacts and environments where agents interact with resources.

The *Continuous Direct Data Dispatcher* and *Action Dispatcher* patterns were tested in a controlled lab environment simulating educational scenarios. Sensors like heart rate monitors, motion detectors, and environmental sensors (lighting, temperature, noise) were used alongside actuators such as smart bands, smart bulbs, and thermostats.

The *Continuous Direct Data Dispatcher Pattern* focuses on collecting and transmitting data from the physical twin to the digital twin (DT). In JaCaMo, this involved creating CArTAgo artifacts to receive and store sensor data in real time. Agents processed this data and integrated it into the DT, ensuring an accurate digital representation of the child and environment. Artifacts acted as intermediaries for efficient data flow.

The *Action Dispatcher Pattern* enables the DT to autonomously execute actions based on analysed data. In JaCaMo, agents interpreted data using a rule-based decision model, determined appropriate actions, and sent commands via CArTAgo artifacts. For example, alerts were sent to a smart band to suggest breaks when fatigue or restlessness was detected, and commands adjusted environmental conditions like lighting or temperature to enhance concentration.

6.2 Discussion

The process of identifying, formalizing, and applying design patterns for the entanglement of DT has revealed several key insights. First, the abstraction of entanglement into the three aspects, connectivity, promptness and association, has proven extremely useful for clarifying design requirements and guiding pattern selection. These dimensions allowed for an objective analysis of DT needs and served as effective heuristics for mapping real-world problems to MAS design patterns.

Second, the use of MAS provided a solid foundation for pattern design. By drawing inspiration from established MAS patterns, such as observer, publish-subscribe, event dispatcher, etc., we ensured that the proposed design patterns are both theoretically grounded and practically applicable.

Third, the decision model introduced in this article proved to be an accessible and practical tool for early-stage architectural planning. Its stepwise structure enabled both novice and experienced architects to reason clearly about entanglement needs and to make informed design decisions.

Finally, the implementation of a DT for managing study sessions using our catalogue of design patterns confirmed that these patterns were not only appropriate in theory but also effective in a practical study case.

6.3 Threats to Validity

Several threats to the validity of this study were identified and are discussed below [33]:

- *Construct Validity*: The definitions of the entanglement aspects are based on the work of Minerva et al. [23]. Although this work is widely accepted (citations: 506; source: Scopus; date: April 2025), it remains subject to interpretation.

- *Internal Validity*: The identification of patterns may have been influenced by the bias of the authors of this paper during the analysis process. However, all patterns were cross-checked against existing MAS literature.
- *External Validity*: The applicability of the patterns was demonstrated in a single case study in an educational setting, so their effectiveness in other settings will need further validation.
- *Implementation Bias*: The use of the JaCaMo [16] facilitated the implementation of the design patterns, however the results may not directly translate to other agent development platforms.

7 Conclusion and Future Work

Respect to **RQ1**, this study contributes a novel catalogue of agent design patterns aimed at supporting the implementation of the entanglement property of DT. The agent design patterns proposed are: *Indirect Observation*, *Continuous Direct Data Dispatcher*, *Action Dispatcher*, *Event-Driven Direct Data Update Pattern*, and *Periodic Direct Data Sending Pattern*. These design patterns allow the development of the three degrees of entanglement: *weak*, *simple* and *strong*.

Regarding **RQ2**, a decision model is proposed to guide software architects in selecting the previous design patterns that best aligns with the entanglement aspects: *connectivity*, *association* and *promptness*. Then, the application of the decision model in a case study validated their practical relevance, demonstrating also how the proposed design patterns can be effectively selected and implemented by using JaCaMo [16].

Finally, future work will focus on further improve the applicability of the proposed catalogue of design patterns extending the catalogue by identifying new design patterns related to other DT properties.

Acknowledgments. This paper has been partially funded by the R + D + i project PID2022-140907OB-I00 funded by MICIU/AEI/10.13039/501100011033 and ERDF, EU. It has also been partially supported by Junta de Comunidades de Castilla-La Mancha/ERDF (SBPLY/24/180501/000020) and by the University of Castilla-La Mancha (2022-GRIN-34436). Elena Pretel holds a FPU21/02679 scholarship from Spanish Ministerio de Educación y Formación Profesional and holds a EST25/00141 scholarship from Spanish Ministerio de Ciencia, Innovación y Universidades.

References

1. Chella, A., Cossentino, M., Gaglio, S., Sabatucci, L., Seidita, V.: Agent-oriented software patterns for rapid and affordable robot programming. *J. Syst. Softw.* **83**(4), 557–573 (2010)
2. Croatti, A., Gabellini, M., Montagna, S., Ricci, A.: On the integration of agents and digital twins in healthcare. *J. Med. Syst.* **44**(9), 1–8 (2020)
3. Deugo, D., et al.: Patterns as a means for intelligent software engineering. In: *Proceedings of the 1999 Conference on Artificial Intelligence* (1999)
4. Dorri, A., Kanhere, S.S., Jurdak, R.: Multi-agent systems: a survey. *IEEE Access.* **6**, 28573–28593 (2018)

5. Dupire, B., Fernandez, E.B.: The command dispatcher pattern. In: 8th Conference on Pattern Languages of Programs (2001)
6. Fernandez-Marquez, J.L., Arcos, J.L., Di Marzo Serugendo, G., Casadei, M.: Description and composition of bio-inspired design patterns: the gossip case. In: 2011 Eighth IEEE International Conference and Workshops on Engineering of Autonomic and Autonomous Systems, pp. 87–96. IEEE (2011)
7. Gayathri, R., Uma, V.: Ontology based knowledge representation technique, domain modeling languages and planners for robotic path planning: a survey. *ICT Express*. **4**(2), 69–74 (2018)
8. Glaser, B.G., Strauss, A.L.: *The Discovery of Grounded Theory: Strategies for Qualitative Research*. Routledge (1999)
9. González-García, J.J., Pretel, E., López-Jaquero, V., Montero, F., González, P.: Smart-Pomodoro: wearable Technology for Improving Time-Management, engagement and focus in children with ADHD. *Int J Hum Computer Inter.* (2025)
10. Gorodetsky, V.I., Kozhevnikov, S.S., Novichkov, D., Skobelev, P.O.: The framework for designing autonomous cyber-physical multi-agent Systems for Adaptive Resource Management. In: *Industrial Applications of Holonic and Multi-Agent Systems: 9th International Conference*, pp. 52–64 (2019)
11. Grieves, M., Vickers, J.: Digital twin: mitigating unpredictable, undesirable emergent behavior in complex systems. In: *Transdisciplinary Perspectives on Complex Systems*, pp. 85–113. Springer International Publishing (2017)
12. Guo, J., Lv, Z.: Application of digital twins in multiple fields. *Multimedia Tools and Appl.* (2022)
13. Heininger, R., Stary, C.: Capturing autonomy in its multiple facets: a digital twin approach. In: *2021 ACM Workshop on Secure and Trustworthy Cyber-Physical Systems*, pp. 3–12. ACM (2021)
14. Huang, J., Koroteev, D.D., Rynkovskaya, M.: Machine learning-based demand response in PV-based smart home considering energy management in digital twin. *Sol. Energy*. **252**(January), 8–19 (2023)
15. ISO 23247-1:2021: Automation systems and integration—Digital twin framework for manufacturing — Part 1: Overview and general principles (2021)
16. JaCaMo: JaCaMo. <http://jacamo.sourceforge.net/>
17. Kitchenham, B., Pearl Brereton, O., Budgen, D., Turner, M., Bailey, J., Linkman, S.: Systematic literature reviews in software engineering – a systematic literature review. *Inf Softw Technol.* **51**(1), 7–15 (2009)
18. Liu, Y., et al.: Agent design pattern catalogue: a collection of architectural patterns for foundation model based agents. *J Syst Softw.* **220**, 112278 (2025)
19. Loh, H.W., Ooi, C.P., Barua, P.D., Palmer, E.E., Molinari, F., Acharya, U.R.: Automated detection of ADHD: current trends and future perspective. *Comput Biol Med.* **146**, 105525 (2022)
20. Luder, A., Cala, A., Zawisza, J., Rosendahl, R.: Design pattern for agent based production system control — a survey. In: *2017 13th IEEE Conference on Automation Science and Engineering (CASE)*, pp. 717–722. IEEE (2017)
21. Marks, P., Muller, T., Vogeli, D., Jung, T., Jazdi, N., Weyrich, M.: Agent design patterns for assistance Systems in Various Domains - a survey. In: *2018 IEEE 14th International Conference on Automation Science and Engineering (CASE)*, pp. 168–173. IEEE (2018)
22. Meszaros, G.: *Archi-Patterns 1: a Process Pattern Language for Defining Architectures* (1997)
23. Minerva, R., Lee, G.M., Crespi, N.: Digital twin in the IoT context: a survey on technical features, scenarios, and architectural models. *Proceedings of the IEEE*. **108**(10), 1785–1824 (2020)

24. Mohamad, R., Deris, S., Ammar, H.H.: Agent design patterns framework for MaSE/POAD methodology. In: IEEE International Conference on Computer Systems and Applications, 2006, pp. 521–528. IEEE (2006)
25. Oluyomi, A., Karunasekera, S., Sterling, L.: An agent design pattern classification scheme: capturing the notions of Agency in Agent Design Patterns. In: 11th Asia-Pacific Software Engineering Conference, pp. 456–463. IEEE (2004)
26. Pretel, E., Moya, A., Navarro, E., López-Jaquero, V., González, P.: Analysing the synergies between multi-agent systems and digital twins: a systematic literature review. *Info Softw Technol.* **174**, 107503 (2024)
27. Pretel, E., Zhinin-Vera, L., Navarro, E., López-Jaquero, V., González, P.: MAS4DT: a novel proposal for developing digital twins following a multi-agent system approach. *J Syst Softw.* **222**, 112344 (2025)
28. Skobelev, P., Tabachinskiy, A., Simonova, E., Lee, T.-R., Zhilyaev, A., Laryukhin, V.: Digital twin of rice as a decision-making service for precise farming, based on environmental datasets from the fields. In: 2021 International Conference on Information Technology and Nanotechnology (ITNT), pp. 1–8. IEEE (2021)
29. Sulaiman Khail, W., Schnizer, P.: Patterns in digital twin development. In: Proceedings of the 29th European Conference on Pattern Languages of Programs, People, and Practices, pp. 1–8. ACM (2024)
30. Tekinerdogan, B., Verdouw, C.: Systems architecture design pattern catalog for developing digital twins. *Sensors.* **20**(18), 5103 (2020)
31. Wedyan, F., Abufakher, S.: Impact of design patterns on software quality: a systematic literature review. *IET Software.* **14**(1), 1–17 (2020)
32. Yun, T.-S., Moon, I.-C., Kim, Y.-C., Jin, K.-S.: Agent-based modeling and simulation on residential population movement patterns: the case of Sejong City. In: 2020 Winter Simulation Conference (WSC), pp. 620–631. IEEE (2020)
33. Zhou, X., Jin, Y., Zhang, H., Li, S., Huang, X.: A map of threats to validity of systematic literature reviews in software engineering. In: 2016 23rd Asia-Pacific Software Engineering Conference (APSEC), pp. 153–160. IEEE (2016)