The Two Faces of Interoperability: Bridging Cyber and Physical Spaces with Digital Twins

The Two Faces of Interoperability: Bridging Cyber and Physical Spaces with Digital Twins

Marco Picone *, Matteo Martinelli *,
Samuele Burattini†, Andrea Giulianelli†, Alessandro Ricci†
* *University of Modena and Reggio Emilia, Italy,* † *University of Bologna, Italy*
{name.surname}@unimore.it, samuele.burattini@unibo.it, andrea.giulianelli7@unibo.it, a.ricci@unibo.it

***Abstract*—The concept of Digital Twins has emerged as a pivotal innovation for bridging the physical and digital realms. This paper investigates the two-faceted nature of interoperability: on the physical side, Digital Twins must integrate with heterogeneous devices and data sources; on the digital side, applications may need to seamlessly interoperate with possibly heterogeneous Digital Twins. We propose a software architecture that addresses this dual interoperability requirement, discuss its alignment with existing standards and showcase it in an industrial use case.**

***Index Terms*—Digital Twins, Industrial Internet of Things, Cyber-Physical Systems, Interoperability**

I. Introduction

The rise of Digital Twins (DTs) has transformed the landscape of Internet of Things (IoT) and Cyber-Physical Systems (CPS), providing a powerful means to bridge the gap between physical assets and digital systems. By creating virtual representations of real-world entities, DTs enable advanced monitoring, simulation, and optimization, unlocking new possibilities for automation, operational activities, and real-time decision-making [1]. Industries such as manufacturing, healthcare, smart cities, and transportation have increasingly adopted DTs to enhance efficiency, reduce operational costs, and improve system reliability [2], [3], [4]. However, some argue the full potential of DTs can only be realized if they achieve seamless interoperability across both physical and digital domains [5], [6]. For the scope of this paper, we consider interoperability as the ability to integrate effectively with other system components, and we argue that in the DT context, such an ability has a twofold nature.

On the one hand, DTs must effectively integrate with a heterogeneous physical world, interacting with diverse IoT devices and systems that operate on different standards, protocols, and architectures. For example machines from different vendors may utilize different communication protocols and interaction patterns. Similarly, in healthcare applications, DTs must integrate various electronic health records across different data sources, from traditional repositories to sensor networks. This *physical* heterogeneity poses a challenge for the development of DTs, as, most often, it is unfeasible to modify the existing conditions and the DT system must be constructed on top of the legacy ones *adapting* to their requirements.

On the other hand, although DTs could work in isolation, they are increasingly being envisioned as part of complex software systems. Thus, DTs needing to integrate with different kinds of services ranging from system automation tools, to data platforms, machine learning pipelines and visualization tools, may expect to face several degrees of *digital* heterogeneity. As for the physical realm, the DT might need to *adapt* to the requirements of the digital ecosystem it is part of. Additionally, the modeling of complex CPS can be tackled through the idea of DT ecosystems [7] which envision multiple connected DTs, each modeling individual entities of the same domain. Such DT ecosystems can benefit from standardized interactions between DTs and with higher-level services.

Tackling such twofold interoperability challenges requires on one side the adoption of common data models, semantic interoperability frameworks, and compliance with industry standards [8]. On the other side, the DT architecture should accommodate physical and digital heterogeneity *by design* to avoid being locked down in those situations where adherence to a single standard can not be enforced.

This paper contributes to the development of DTs in heterogeneous IoT environments by: (*i*) examining the dual nature of interoperability in DTs through the lens of software engineering principles; (*ii*) proposing a general-purpose modular architecture for DTs based on the concept of physical and digital *adapters* which encapsulate the responsibility of mediating the interaction between the DT and other physical and digital systems; and (*iii*) advocating for the generation of *Physical Asset Descriptions (PADs)* and *Digital Twin Descriptions (DTDs)* to support the decoupling between the DT model, physical systems and digital applications. By addressing these concerns at the architectural level, our approach enhances the interoperability and re-usability of DT components in cyber-physical applications.

The paper is structured as follows: In section II, we provide the relevant background on DT modeling and further motivate the interoperability challenges that arise when designing a DT system. In Section III, we discuss our approach, distinguishing the roles and responsibilities of the adapters dedicated to achieve physical and digital interoperability. In Section IV, we discuss how Web of Things (WoT) standards [9] align with our proposal and complement it, showing how a combined approach can enhance DT interoperability in both digital and physical layers. Finally, to showcase our proposal, we illustrate how the presented DT design can achieve interoperability in an ecosystem of DTs modeling a manufacturing scenario through an Industrial Microfactory in Section V.

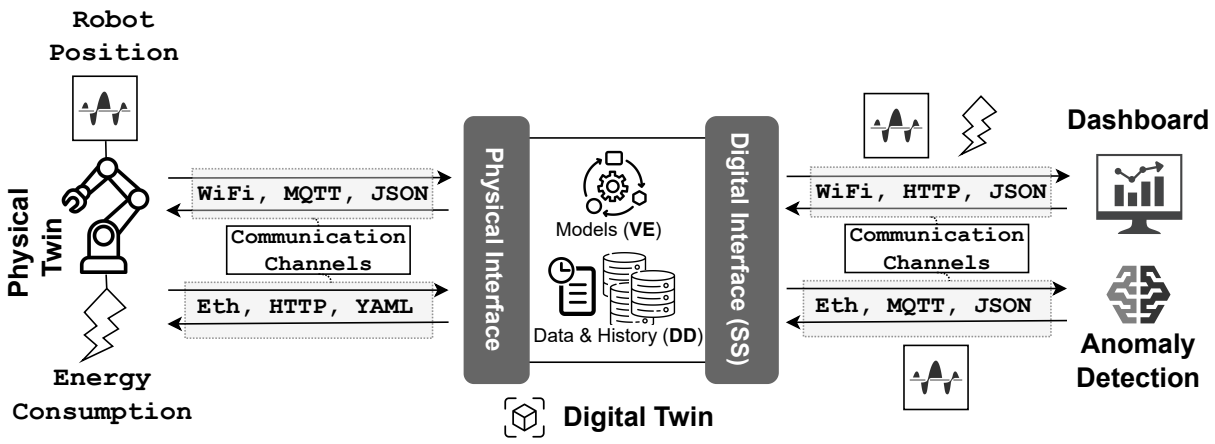


Fig. 1: A DT bridging physical and digital spaces via communication interfaces enabling data exchange and supporting models.

II. Digital Twin Modeling

Several approaches have been proposed to model DTs in different contexts [10]; hence, providing a comprehensive review is out of the scope of this work. Since our proposal is grounded on the abstract DT model originally presented in [7], in this section, we describe its main features. We further recall other proposals from the scientific [11], [12], [13] and standardization communities [8] aligning it with the work presented in [7] to provide a background on DT modeling.

A. Digital Twin Abstract Architecture

Among the most recognized models for DTs, Tao's 5D model [12], defines five fundamental dimensions a DT must have: the physical entity (PE), virtual entity (VE), connection (C), data (DD) and services (SS) dimensions. The model can be mapped into an abstract software architecture (which is a generalization of the one proposed in [7]) structured with three main components, as depicted in Figure 1: i) *Physical Interface (PI)*: takes care of the connection (C) and synchronization with the physical entity (PE); ii) *Model (M)*: data gathered by the PI is passed to the DT models, which elaborate it, implementing the virtual entity (VE) absolving to the digitalization purpose of the DT and storing relevant data (DD); and iii) *Digital Interface (DI)*: exposes the DT model computation results as well as endpoints for action and query requests to external entities as services (SS), creating a connection (C) between the DT and digital applications.

Internally, a process identified as *twinning* or *shadowing* [14], [7] handles the transformation of data received from the PI, feeding the models that characterize the DT. The result of the shadowing process is to compute the state of the DT, which can be represented as a set of: i) **Properties**: labeled data that change with the evolution of the physical counterpart state; ii) **Events**: non-persistent signals captured by the information gathered from the associated physical asset; iii) **Relationships**: capturing the existing and dynamic connections among physical assets within the system, mirroring them between the corresponding DTs; and iv) **Actions**: the set of possible operations that the DT allows to invoke on behalf of the physical counterpart, to either send feedback to the physical entity or exploit a service exposed by the DT [7]. State updates can then be communicated to applications through the DI, fulfilling the objective of the DT in providing an up-to-date representation of the Physical Twin (PT). The digital invocation of actions on the DT is also converted to the appropriate physical actuation through the shadowing process, achieving bi-directional synchronization [7].

B. Digital Twin Lifecycle

A DT is a "living" software entity: The concept of DT lifecycle, as introduced in [7], covers the various phases a DT undergoes from creation to deactivation, focusing on its synchronization with the PT. The phases are illustrated in Figure 2. The DT transitions from the `Unbound` phase, where internal modules are prepared, to the `Bound` phase once it successfully binds with the PT. In the `Synchronized` phase, the DT maintains an up-to-date digital replica of the PT. If synchronization fails, the DT enters the `Out of Sync` phase until issues are resolved. When the DT is no longer required, it transitions to the `Done` phase, then to `Stopped`. Errors or reboots may lead back to the `Unbound` phase.

The transition from the Unbound to Bound phase in the DT lifecycle remains an open research area: a key challenge is identifying whether all the capabilities of the PT relevant to support the DT's behavior are available so that the shadowing process can start. We further elaborate on how to tackle this challenge with our contribution in Section III-A.

III. Engineering Digital Twin Interoperability

Figure 1 represents a DT implemented following the abstract architecture presented in Section II in an exemplary deployment setting. It highlights the role of the DT as a bridge between *physical* devices and *digital* applications.

At an appropriate abstraction level, a PT may be digitalized by obtaining and sending data through possibly *several* different `Communication Channels`, which encompass network protocols, data formats, and physical connections.

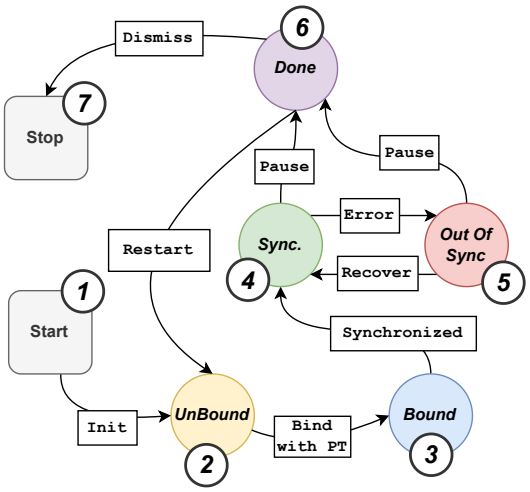


Fig. 2: Digital Twin lifecycle phases and transactions.

The PI manages interactions with the PT, integrating such channels. Similarly, on the digital side, the DT supplies data and insights to applications via the DI, adapting to different protocols and representations.

We argue that to ensure flexibility and adaptability, it is essential that the core of the DT, including its models and implemented behaviors, remains decoupled from the heterogeneous physical and digital communication channels.The DT's shadowing process should focus solely on understanding the available physical data, receiving and transmitting telemetry information, and executing action requests, with no concern for the underlying implementation details. This separation of concerns should be supported by the DT architecture and achieved through a modular design of the DT components, which has been proven effective [13], [15] as it strengthens extensibility, interoperability and maintainability with respect to monolithic approaches.

In the remainder of this section, we present our proposal for the design the DT's PI and DI, refining and extending the conceptual model originally proposed in [7] with a concrete implementation based on the concept of modular *adapters*.

A. Physical Interoperability

One of the main challenges in facilitating effective communication through the PI is the wide variety of communication protocols and data formats used by PTs. While it can be the case that one PT is directly sending all the data concerning its digitalization through only one channel, it is far more common to build a DT aggregating different sources of information [16]. IoT protocols often differ considerably based on the manufacturer, device type, or application domain, making it difficult for the DT software to integrate diverse assets.

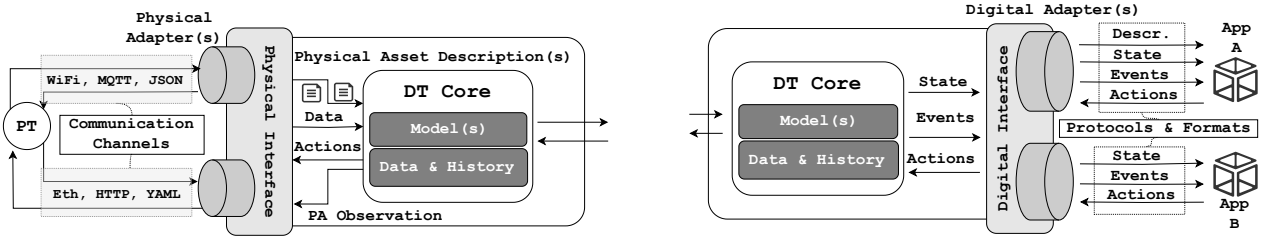
We argue that the modularity of the PI, which encapsulates these responsibilities, is a crucial factor in the design and implementation of interoperable DTs. Hence, we propose the PI to be designed as composed of multiple *Physical Adapters (PAs)*: specialized modules capable of interacting with the PT using diverse protocols and data formats. As shown in Figure 3a, each PA is responsible for mediating the bi-directional interaction through a single communication channel, simplifying the design and reusability of the component, and making it configurable to adapt to different application contexts. The responsibility of the PI becomes then to manage the different PAs and make sure the DT model can accurately interpret, process, and leverage the data generated by the physical world to create the digital replica and implement its behaviors. Note that although we borrow the terminology of PA from [7], where it is originally introduced, our proposal differs significantly. The *Physical Asset Adapter* proposed in [7] is infact a conceptual monolithic component, whereas in our concrete proposal, we consider a PA as a single-responsibility module of a potentially more complex PI.

1) Generating Physical Asset Descriptions: To facilitate the process of managing different PAs, we introduce the idea of generating a *PAD*: a representation of the capabilities made available by a communication channel in terms of *properties*, *actions*, *events* and *relationships* that characterize the associated PT. Each PA, since it encapsulates the characteristics of a channel, can generate the corresponding PAD, effectively decoupling the asset's functionality from the specific communication protocols used. The implementation of the PAD generation can be challenging due to the varying capabilities of different communication protocols. For instance, protocols like CoAP[17] often come with built-in description and discovery functionalities, which can simplify the process of creating a PAD by providing standardized representations of the physical asset's capabilities and behaviors. On the other hand, protocols like MQTT may require developers to add additional information manually to define how information is structured and exchanged, as they do not natively include asset metadata [18]. In the case of custom or proprietary protocols, the challenge becomes even more pronounced, as these protocols are tailored to specific systems and may lack any standardization or descriptive capabilities.

In all the aforementioned scenarios, the mechanism of PAD generation offers a way to encode knowledge about the PT bridging the gap by interpreting and extracting relevant information from the protocol used in the associated communication channels. Confining this complexity within the PI allows the DT model to be agnostic with regard to the complexity of the underlying physical world.

The PAD allows the PI to discover, extract, and manage asset information and present it to the model that can choose which ones are relevant for the implementation of the DT behavior. For example, to create the DT of a temperature sensors streaming binary data on MQTT, the PI could be composed of a generic MQTT adapter, configured to correctly parse the payload as a decimal number, and generate a PAD which advertise the available temperature property to the DT model as a Celsius value.

2) Physical Asset Descriptions in the DT Lifecycle: The modular design of the PI has an impact on the DT lifecycle presented in Figure 2. Specifically, it tackles the open challenge of transitioning from the `Unbound` to the `Bound` state.



(a) Physical Interface (b) Digital Interface

Fig. 3: DT interfaces and their responsibilities to enable and support cyber-physical interoperability.

When a PA successfully connects to the PT's channel and starts receiving data from it, it can send the generated PAD to the DT model. The generation of the PAD can be used as a synchronization step to signify that the PA is successfully connected to the PT. The DT model is then responsible for collecting the different PADs, assessing whether all the relevant information to start computing the DT state is available and, hence, moving on to the `Bound` phase. This mechanism enables managing the DT behavior consistently, even with the additional complexity of the modular PI design.

B. Digital Interoperability

As DTs are meant to bridge between the physical and digital spaces, interoperability is not only a matter of interfacing with heterogeneous devices, but also other digital applications. Indeed, despite their initial conception as vertical silos, the concept of Digital Twins Ecosystems [7], [19] is emerging to support the the digitalization of complex scenarios through a combination of several DTs. In this context, DTs can be used *as-a-service* [20] by other applications implementing the business logic by spanning across different software entities.

To foster interoperability in ecosystems, DTs must then expose either a standardized general purpose Digital Interface (DI) that can serve different applications or – following the same design principles adopted to address physical interoperability – have a modular interface that can satisfy the different needs of different applications as shown in Figure 3b.

As for PA we borrow the terminology from [7] and consider to have a DI composed of modular Digital Adapter (DA). We thus refine the original abstract concept of DA to represent a modular component of a more complex DI. Using the concept of *DA*, the DI can expose the DT state and services supporting multiple data formats and interaction patterns. This can be beneficial for integrating it with applications and, especially, legacy systems. The existence of legacy applications usually implies having little control over the integration requirements, making having more flexible DTs beneficial to better integrate with the application requirements. Using several DA, a DT could, for instance, support both request-response and publish-subscribe mechanisms to access its current and previous states, support different query languages to access the same data store, expose its current state using different representation formats, etc. This would make the development of the application simpler because adding an application-specific DA won't require intervention in the underlying physical system. Additionally, the developed application would be more robust and stable since it would depend only on the DT, and changes to the physical configuration of the PT (e.g., software updates, sensor replacement, network reconfigurations) would have no impact on the application software. Even if the PT were to change, (e.g., a software update on a robot changes the telemetry format) the DI of the DT could stay the same, as the changes would occur within the boundaries of the PI and DT model. Through this mechanism, DTs effectively achieve their bridging role, shielding applications from the complexity of physical deployments.

1) Describing Digital Interfaces: A further level of interoperability is possible when allowing DTs to describe their DI, advertising capabilities and available communication channels that applications can exploit. A DT could then use a *DTD*, which, similarly to the PAD, can represent the features of the DT to its observers. The idea of exposing DTDs is somewhat present in the major platforms supporting the creation of DT ecosystems, such as Microsoft's Azure Digital Twins[1] and Eclipse Ditto[2] and is advocated by research works on interoperability in DT ecosystems [8], [21].

The way such descriptions are implemented may differ significantly, but essentially, they should at least allow representing the DT features and APIs to access them. Notably, differently from the PAD, the DTD is targeted to external consumers, hence, it should preferably adhere to standard formats and representations to be useful in practice in achieving interoperability. To this end, using Linked Data principles[3] could be a possible solution to implement standard machine-readable DTDs [22].

IV. Web of Things: enabling DT Interoperability

Tackling interoperability in the IoT context is not a novel issue. Before the emergence of DTs several solutions have attempted to solve it through the definition of standards [23].

In this section, we analyze the set of standards produced for the WoT [9] as they share the goal of our proposal of adopting uniform API and data format descriptions to hide low-level complexities and promote interoperability.

[1]https://azure.microsoft.com/en-us/products/digital-twins

[2]https://eclipse.dev/ditto/

[3]Original definition of the Linked Data principles: https://www.w3.org/DesignIssues/LinkedData.html

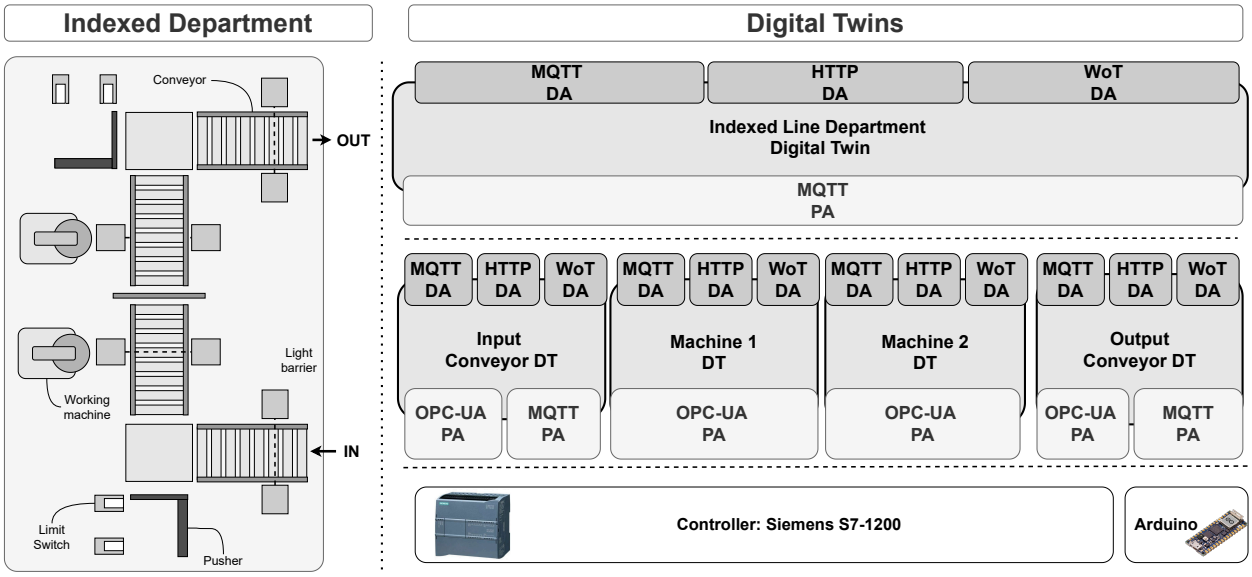


Fig. 4: The Digital Twin ecosystem architecture of the microfactory industrial system.

In the context of the WoT, *Things* represent abstractions over physical or virtual entities that participate in the WoT [9]. WoT Things are described in terms of metadata and interfaces by the WoT Thing Description (TD) [24]. A TD provides a machine-readable blueprint for describing metadata and *interaction affordances* of a physical object [24].

Despite their similarities, PADs and TDs serve distinct purposes. A TD provides a structured description of a physical twin's available interactions to external consumers, detailing protocols and interaction possibilities. In contrast, a PAD is designed for internal use within the DT modules, decoupling the twin's core from the complexities of communication channels. Its primary role is to facilitate the discovery of available resources and capabilities on the PT after establishing a connection with the physical counterpart, providing a structured description of its functionalities and data. This description is directly interpretable by the DT model while remaining independent of the physical characteristics, allowing, for instance, the reuse of the same model for similar PTs employing different communication approaches.

In those context in which WoT standards are applicable, the devices' TDs can be automatically mapped to PADs, streamlining the DT creation process. This convergence between WoT and DT architectures has the potential to accelerate the development and deployment of DT solutions by promoting standardization and reusability. Nevertheless, the more general concept of PAD can be tailored also to those scenarios in which WoT is not applicable. In those cases the responsibility falls back to development (or configuration) of a PA for a specific device to encode the necessary knowledge for the generation of the PAD.

WoT TDs can also be used as DTDs. The WoT architecture actually lists DTs as one of the possible deployment patterns, with a DT mediating the interaction with a PT behind a WoT interface [4]. This is especially useful when devices are not WoT compatible or can not be made so due to other constraints, essentially retrofitting the capabilities of the devices with a WoT compatible interface. Adhering to REST constraints [25] – the architectural style of the Web – and specifically to the HATEOAS and self-descriptive principles, a TD-based DTD facilitates the discovery of the DT model and services. Its machine-readable nature ensures interoperability for applications and consumers. In particular, a TD-based DTD facilitates the inclusion of DTs in WoT mashups, promoting DTs as valid WoT Things usable by WoT Consumers.

Despite its flexibility, using of TD for DTs presents several limitations in capturing the specific characteristics of DTs that distinguish them from *Things* [22]. A path to address these limitations could be extending TDs or supporting additional descriptions specifically for DTs in DT ecosystem [21].

V. Interoperability in a Manufacturing Use Case

The proposed DT architectural approach is showcased using the *Fischertechnik Training Factory Industry 4.0*[5] Indexed-Line Station, controlled by a Siemens PLC that publishes real-time data via OPC-UA, integrated with Arduino RP2040 boards[6] for accelerometer data collection and processing and communicating over MQTT[7].

The scenario emulates a production line, with a DT system deployed to monitor production and measuring the overall efficiency. The DT ecosystem architecture is depicted in Figure 4. Four machine-level DTs are deployed, each with a PI supporting the relevant communication protocols to interact with

[4]https://www.w3.org/TR/2023/REC-wot-architecture11-20231205/#digital-twins

[5]Fischertechnik Industry & Universities: https://www.fischertechnik.de/en/products/industry-and-universities

[6]Arduino RP2040: https://docs.arduino.cc/hardware/nano-rp2040-connect/

[7]MQTT Protocol: https://mqtt.org/

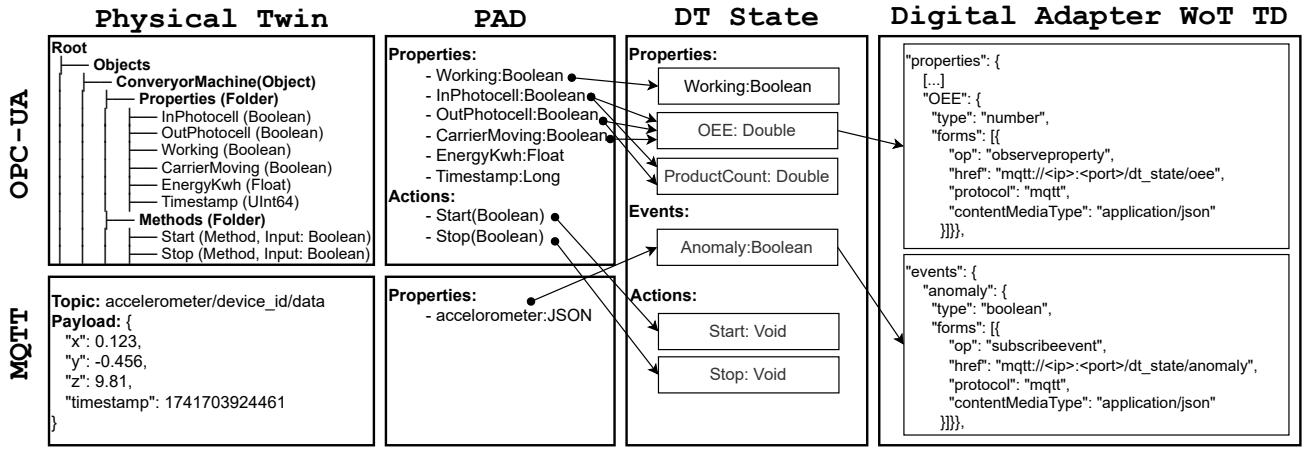
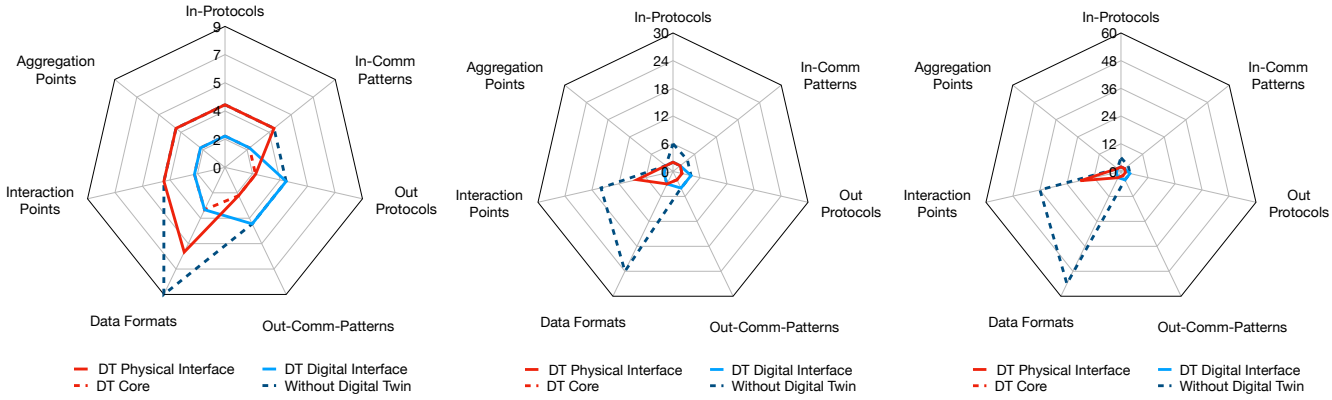


Fig. 5: The transition from physical to digital representation via the PAD, the DT State, and the DTD for external consumers.



(a) Machine Digital Twin (b) Station (Indexed-Line) Digital Twin (c) Factory Digital Twin

Fig. 6: Schematic representation of cyber-physical complexity and its impact on various components.

the different machines, namely through MQTT and OPC-UA[8]. These DTs track machine states, monitor performance, and coordinate production by processing data from the physical system and digital action requests. An additional department-level DT aggregates data from machine-level DTs, extracting system metrics and exposing department-level actions.

All the DTs are built using the White Label Digital Twins (WLDT) open-source framework[9]: a Java-based multithreaded stack, facilitating the definition of DT and supporting the implementation of shadowing and digitalisation processes. The frameworks implements our proposal of physical and digital adapters that we exercise in this example.

DT of composed physical assets, such as the *Indexed Line Department* DT (see Figure 4, top) use lower-level DTs as data sources. Notably, the physical and digital adapters that implement MQTT, OPC-UA, and HTTP protocols, along with WoT over HTTP for serving TDs, are reusable across different instances. These adapters, through configurable parameters, enable effective communication and integration within both digital and physical environments.

Machine-level DTs provide real-time data on their physical counterparts, such as workpiece presence, operational state (e.g., idle, working, waiting, or broken), and performance KPIs like Overall Equipment Efficiency (OEE)[10], calculated from production rate, uptime, and downtime. At the department level, DTs representing the Indexed Departments assess performance by computing Weighted OEE [26], [27], derived from individual machine DTs, and track throughput, defined as processed pieces per unit time. Throughput is determined by monitoring entry and exit events within the department, utilizing relationships captured by the associated DTs.

Upon startup, each DT connects to its data source, publishes the PAD of the physical asset, and interacts with the PLC or Arduino to collect and send data, enabling digitalization and interoperability among heterogeneous equipment.

Figure 5 schematically illustrates the evolution of data and representation from the physical to the digital realms through the adoption of the proposed DT architectural approach, PAD integration, and WoT TDs used as DTDs on the digital side, specifically for the output conveyor of the target production line. On the physical side, data is structured using OPC-UA

[8]OPC-UA: https://opcfoundation.org/about/opc-technologies/opc-ua/
[9]WLDT framework: https://wldt.github.io
[10]OEE: https://www.oee.com/

with a hierarchical organization, containing telemetry data and action methods mapped to specific data types. Accelerometer data is transmitted over MQTT on a specific topic with a JSON payload containing axis acceleration values.

The PADs act as an intermediary, translating data from the PT into a format suitable for the DT core, decoupling it from the complexity of interacting with the PT and enabling discovery, understanding, and utilization of available data and methods. Using the information from the PADs, the DT model computes the DT's state with target properties, events, and actions, based on either one-to-one matching of physical characteristics or augmented by combining multiple physical properties into computed DT properties or events, such as computing the value of the OEE property or triggering anomaly detection events based on accelerometer data.

Finally, the DA of the DT leverages WoT TD to describe the DT's data and functionalities through a standardized, interoperable, and machine-readable approach. The TD specifies how these can be accessed via protocols and data formats. This approach enables effective interoperability from the physical to the digital, using a modular and decoupled structure through DTs in the target industrial use case.

A. Mapping Cyber-Physical Complexity

In our experimental analysis, we also aim to measure the impact of digitization and responsibility decoupling by comparing the system complexity in scenarios with and without DT adoption and also with respect to the different DT's architectural layers. This allows us to assess the benefits of a modular, structured DT-based approach, particularly in terms of interoperability and heterogeneity management. To quantify how effectively our approach manages cyber-physical heterogeneity, we adopt the *Cyber-Physical Complexity Index (CP-CI)* proposed in [28], [29]. The CP-CI quantifies the complexity of a cyber-physical application based on: i) *Required Protocols (p)*: communication protocols needed for device interaction; ii) *Communication Patterns (c)*: interaction models (e.g., Publish/Subscribe, Request/Response); iii) *Data Formats (d)*: required data representations or serialization methods; iv) *Interaction Points (n)*: modules or services involved in data exchange; and v) *Aggregation Points (a)*: levels of abstraction for physical data (e.g., merging data from multiple machines). Each criterion is weighted with a *Criteria Importance Factor (CIF)* from 1 (low) to 3 (high), indicating its impact on development, deployment, and maintenance.

Focusing on DT interoperability, we extended the CP-CI to include both inbound and outbound interfaces, essential for bridging the physical and cyber layers with differing requirements. This extension enables a more precise assessment of the complexity in managing interoperability across system boundaries. The CP-CI was applied not only to the entire DT but also to its internal layers: Physical Interface (PI), Core, and Digital Interface (DI). These results were compared to those of a monolithic external application that directly handles business logic and interoperability, bearing the full complexity, highlighting the advantages of the modular DT design. High importance ($CIF = 3$) was assigned to managing heterogeneous data formats (d) and aggregation points (a), as they are crucial for creating interoperable data models. Medium importance ($CIF = 2$) was given to protocol diversity (p) and interaction with multiple entities (n), which become more significant as systems scale. Low importance ($CIF = 1$) was assigned to multiple communication patterns (c), as their concurrent use is standard in distributed applications.

The graphs presented in Figure 6 display the CP-CI values obtained across different levels of deployment: a single machine, a station in the factory (such as the indexed line, where multiple machines and their respective DTs are managed and composed into a single, unified DT), and the entire factory, which includes 9 machines, 2 composed DTs, and one overarching DT for the entire factory, all deployed following the proposed modular and interoperable approach. For the machines analyzed, the involved protocols (p) and communication patterns (c) are both set to 2, as we utilize MQTT and OPC-UA with Pub/Sub and Request/Response interactions. On the digital side, both MQTT and HTTP are employed. Regarding data formats (d), each machine manages 2 distinct formats, accounting for both PLC data and Arduino accelerometer information. Furthermore, the interaction points (n) and aggregation points (a) are both 2, reflecting the aggregation and interaction with two different sub-entities for each machine and its corresponding DT.

The results for single machines (Figure 6a) highlight the critical role of the PI in the DT architecture. As a decoupling layer, the PI isolates physical-world complexities from the DT core, managing priorities like protocols, data formats, interaction points, aggregation strategies, and communication patterns. This allows the DT core to focus on uniform data, processed through the interface via PAD descriptions and payload transformations, without dealing with heterogeneous physical entities. Consequently, the core interacts only with its internal communication pattern and consistent interaction points, regardless of the physical environment's configuration. A similar approach is applied to the DI, which is decoupled from physical complexities and adapts to external digital systems, ensuring modularity and reuse.

In contrast, systems without a DT must embed all heterogeneity management into a single application, burdening it with business logic, protocol diversity, and data format interoperability. This increases complexity and reduces scalability, as reflected by the complexity index parameters.

As system architecture scales from individual machines to stations, production lines (Figure 6b), or entire factories (Figure 6c), protocol diversity stabilizes while the number of interaction points and data formats increases due to the need for managing heterogeneous components. Our complexity measure shows that systems at the station or factory level experience a significant rise in complexity, driven by interaction points, aggregation nodes, and data format diversity. These results emphasize the benefits of a modular and interoperable approach. By encapsulating complexity within DT modules and using interoperable representations, developers can sim-

plify application development. In contrast, monolithic systems struggle with managing interoperability and system evolution as integration requirements grow.

VI. Conclusion

Digital Twins are transforming IoT and cyber-physical systems in various industrial settings, but their successful implementation requires bridging heterogeneous physical and digital spaces. This paper addresses the challenges in designing and deploying DTs for industrial environments, focusing on interoperability with IoT devices, protocols, data formats, and legacy system requirements. Building on state-of-the-art models [12], [7], we propose a DT architecture that enhances adaptability, promotes code reuse, and ensures interoperability. We base our contribution around modular physical and digital *adapters*, single-responsibility components that manage communication complexities, keeping the DT model isolated from technical details. Using Physical Asset Description (PAD) and Digital Twin Description (DTD), we demonstrate how this approach separates responsibilities, enabling DTs to bridge physical and digital spaces. Instead of proposing novel formats for these description, we choose to align them with existing standards serving as a base for their implementation in applicable contexts. The proposed approach has been applied and showcased in a real-world Industrial Microfactory, highlighting the ease of integrating various industrial standards. Future work will focus on improving DT interoperability, integrating standards, adding semantic annotations, and addressing DT composition challenges.

Acknowledgments

This work is supported by the European Union - NextGenerationEU - under PRIN 2022, Project ID: 20223N7WCJ and CUP: E53D23007770001, Project Title: TWINKLE - digital TWIN continuum: a Key enabler for pervasive cyber-physicaL Environments.

References

[1] R. Minerva, G. M. Lee, and N. Crespi, "Digital twin in the iot context: A survey on technical features, scenarios, and architectural models," *Proc. IEEE*, vol. 108, no. 10, pp. 1785–1824, 2020.

[2] V. Damjanovic-Behrendt and W. Behrendt, "An open source approach to the design and implementation of digital twins for smart manufacturing," *International Journal of Computer Integrated Manufacturing*, vol. 32, p. 366–384, May 2019.

[3] A. Ricci, A. Croatti, and S. Montagna, "Pervasive and connected digital twins - A vision for digital health," *IEEE Internet Comput.*, vol. 26, no. 5, pp. 26–32, 2022.

[4] N. Mohammadi and J. E. Taylor, "Smart city digital twins," in *2017 IEEE Symposium Series on Computational Intelligence, SSCI 2017, Honolulu, HI, USA, November 27 - Dec. 1, 2017*. IEEE, 2017, pp. 1–5.

[5] S. Acharya, A. A. Khan, and T. Päivärinta, "Interoperability levels and challenges of digital twins in cyber–physical systems," *Journal of Industrial Information Integration*, vol. 42, p. 100714, Nov. 2024.

[6] R. Klar, N. Arvidsson, and V. Angelakis, "Digital twins' maturity: The need for interoperability," *IEEE Systems Journal*, vol. 18, no. 1, p. 713–724, Mar. 2024.

[7] A. Ricci, A. Croatti, S. Mariani, S. Montagna, and M. Picone, "Web of digital twins," *ACM Trans. Internet Technol.*, vol. 22, no. 4, nov 2022.

[8] ETSI Specialist Task Forces (STF) 628, "Smartm2m; digital twins communication requirements," Technical Specification, European Telecommunications Standards Institute (ETSI), TS TS-103-845-V1.1.1, February 2024.

[9] M. Lagally, R. Matsukura, M. McCool, K. Toumura, K. Kajimoto, T. Kawaguchi, and M. Kovatsch, "Web of Things (WoT) Architecture," World Wide Web Consortium, W3C Recommendation, 2023. [Online]. Available: https://www.w3.org/TR/2023/REC-wot-architecture11-20231205/

[10] F. Tao, B. Xiao, Q. Qi, J. Cheng, and P. Ji, "Digital twin modeling," *Journal of Manufacturing Systems*, vol. 64, pp. 372–389, 2022.

[11] F. Tao, M. Zhang, Y. Liu, and A. Nee, "Digital twin driven prognostics and health management for complex equipment," *CIRP Annals*, vol. 67, no. 1, pp. 169–172, 2018.

[12] Q. Qi, F. Tao, T. Hu, N. Anwer, A. Liu, Y. Wei, L. Wang, and A. Nee, "Enabling technologies and tools for digital twin," *Journal of Manufacturing Systems*, vol. 58, pp. 3–21, 2021, digital Twin towards Smart Manufacturing and Industry 4.0.

[13] P. Bellavista, N. Bicocchi, M. Fogli, C. Giannelli, M. Mamei, and M. Picone, "Requirements and design patterns for adaptive, autonomous, and context-aware digital twins in industry 4.0 digital factories," *Computers in Industry*, vol. 149, p. 103918, 2023.

[14] R. Saracco, "Digital twins: Bridging physical space and cyberspace," *Computer*, vol. 52, no. 12, pp. 58–64, 2019.

[15] P. Bellavista, N. Bicocchi, M. Fogli, C. Giannelli, M. Mamei, and M. Picone, "Exploiting microservices and serverless for digital twins in the cloud-to-edge continuum," *Future Gener. Comput. Syst.*, vol. 157, pp. 275–287, 2024. [Online]. Available: https://doi.org/10.1016/j.future.2024.03.052

[16] Q. Qi and F. Tao, "Digital twin and big data towards smart manufacturing and industry 4.0: 360 degree comparison," *IEEE Access*, vol. 6, pp. 3585–3593, 2018.

[17] Z. Shelby, K. Hartke, and C. Bormann, "The constrained application protocol (coap)," Internet Requests for Comments, RFC Editor, RFC 7252, June 2014.

[18] O. Standard, "Mqtt," *OASIS*, 2019. [Online]. Available: https://docs.oasis-open.org/mqtt/mqtt/v5.0/mqtt-v5.0.html

[19] J. Kendall, *National Digital Twin: Integration Architecture Pattern and Principles*. UK: CDBB, 2021. [Online]. Available: https://www.repository.cam.ac.uk/handle/1810/321334

[20] Y. Liu, S. Ong, and A. Nee, "State-of-the-art survey on digital twin implementations," *Advances in Manufacturing*, vol. 10, no. 1, pp. 1–23, 2022.

[21] A. Giulianelli, S. Burattini, A. Ciortea, and A. Ricci, "Engineering interoperable ecosystems of digital twins: A web-based approach," in *Proceedings of the ACM/IEEE 27th International Conference on Model Driven Engineering Languages and Systems, MODELS Companion 2024, Linz, Austria, September 22-27, 2024*, M. Wimmer, A. Egyed, B. Combemale, and M. Chechik, Eds. ACM, 2024, pp. 476–485.

[22] S. Burattini, A. Zimmermann, M. Picone, and A. Ricci, "Towards linked data for ecosystems of digital twins," in *Proceedings of the ACM/IEEE 27th International Conference on Model Driven Engineering Languages and Systems, MODELS Companion 2024, Linz, Austria, September 22-27, 2024*. ACM, 2024, pp. 332–337.

[23] E. Lee, Y.-D. Seo, S.-R. Oh, and Y.-G. Kim, "A survey on standards for interoperability and security in the internet of things," *IEEE Communications Surveys & Tutorials*, vol. 23, no. 2, pp. 1020–1047, 2021.

[24] S. Kaebisch, M. McCool, E. Korkan, T. Kamiya, V. Charpenay, and M. Kovatsch, "Web of Things (WoT) Thing Description," World Wide Web Consortium, W3C Recommendation, 2023. [Online]. Available: https://www.w3.org/TR/2023/REC-wot-thing-description11-20231205/

[25] R. T. Fielding and R. N. Taylor, "Principled design of the modern web architecture," *ACM Trans. Internet Techn.*, vol. 2, no. 2, pp. 115–150, 2002.

[26] R. Gamberini, L. Galloni, F. Lolli, and B. Rimini, "On the analysis of effectiveness in a manufacturing cell: A critical implementation of existing approaches," *Procedia Manufacturing*, vol. 11, pp. 1882–1891, 1 2017.

[27] S. Nakajima, *Introduction to TPM: total productive maintenance*. Productivity Press, 1995.

[28] M. Lippi, M. Martinelli, M. Picone, and F. Zambonelli, "Enabling causality learning in smart factories with hierarchical digital twins," *Computers in Industry*, vol. 148, p. 103892, 2023.

[29] G. Lombardo, M. Picone, M. Mamei, M. Mordonini, and A. Poggi, "Digital twin for continual learning in location based services," *Engineering Applications of Artificial Intelligence*, vol. 127, p. 107203, 2024.