ETSI TS 103 846 V1.1.1 (2024-08)

SmartM2M;
Digital Twins: Functionalities and communication Reference Architecture

Reference

DTS/SmartM2M-103846

Keywords

digital twins, IoT, oneM2M, ontology

ETSI

650 Route des Lucioles
F-06921 Sophia Antipolis Cedex - FRANCE

Tel.: +33 4 92 94 42 00 Fax: +33 4 93 65 47 16

Siret N° 348 623 562 00017 - APE 7112B
Association à but non lucratif enregistrée à la
Sous-Préfecture de Grasse (06) N° w061004871

Contents

Intellectual Property Rights

Essential patents

IPRs essential or potentially essential to normative deliverables may have been declared to ETSI. The declarations pertaining to these essential IPRs, if any, are publicly available for **ETSI members and non-members**, and can be found in ETSI SR 000 314: *"Intellectual Property Rights (IPRs); Essential, or potentially Essential, IPRs notified to ETSI in respect of ETSI standards"*, which is available from the ETSI Secretariat. Latest updates are available on the ETSI Web server (https://ipr.etsi.org/).

Pursuant to the ETSI Directives including the ETSI IPR Policy, no investigation regarding the essentiality of IPRs, including IPR searches, has been carried out by ETSI. No guarantee can be given as to the existence of other IPRs not referenced in ETSI SR 000 314 (or the updates on the ETSI Web server) which are, or may be, or may become, essential to the present document.

Trademarks

The present document may include trademarks and/or tradenames which are asserted and/or registered by their owners. ETSI claims no ownership of these except for any which are indicated as being the property of ETSI, and conveys no right to use or reproduce any trademark and/or tradename. Mention of those trademarks in the present document does not constitute an endorsement by ETSI of products, services or organizations associated with those trademarks.

DECT™, **PLUGTESTS™**, **UMTS™** and the ETSI logo are trademarks of ETSI registered for the benefit of its Members. **3GPP™** and **LTE™** are trademarks of ETSI registered for the benefit of its Members and of the 3GPP Organizational Partners. **oneM2M™** logo is a trademark of ETSI registered for the benefit of its Members and of the oneM2M Partners. **GSM**® and the GSM logo are trademarks registered and owned by the GSM Association.

Foreword

This Technical Specification (TS) has been produced by ETSI Technical Committee Smart Machine-to-Machine communications (SmartM2M).

Modal verbs terminology

In the present document "**shall**", "**shall not**", "**should**", "**should not**", "**may**", "**need not**", "**will**", "**will not**", "**can**" and "**cannot**" are to be interpreted as described in clause 3.2 of the ETSI Drafting Rules (Verbal forms for the expression of provisions).

"**must**" and "**must not**" are **NOT** allowed in ETSI deliverables except when used in direct citation.

1 Scope

The present document has several key goals. It aims to design and present a clear blueprint for how Digital Twins (DTs) can be designed and structured, how they work and communicate with respect to both physical and digital entities. The present document relies on the insights and information coming from use cases present in ETSI TR 103 844 [i.2] and established DT functionalities and requirements presented in ETSI TS 103 845 [i.3]. The present document defines how DTs can be designed through the presentation of a reference blueprint architecture with a specific focus on communication, discoverability and interoperability aspect. It also sets guidelines for DT adoption and applicability analysing how they can be deployed and monitored over time. Furthermore, the present document specifies the definition and communication aspects for DTs, defining their fundamental characteristics and the requirements for their communications and interoperability, through edge-cloud continuum deployments and with respect to their Physical and Digital Interfaces.

2 References

2.1 Normative references

References are either specific (identified by date of publication and/or edition number or version number) or non-specific. For specific references, only the cited version applies. For non-specific references, the latest version of the referenced document (including any amendments) applies.

Referenced documents which are not found to be publicly available in the expected location might be found at https://docbox.etsi.org/Reference.

NOTE: While any hyperlinks included in this clause were valid at the time of publication, ETSI cannot guarantee their long-term validity.

The following referenced documents are necessary for the application of the present document;

Not applicable.

2.2 Informative references

References are either specific (identified by date of publication and/or edition number or version number) or non-specific. For specific references, only the cited version applies. For non-specific references, the latest version of the referenced document (including any amendments) applies.

NOTE: While any hyperlinks included in this clause were valid at the time of publication, ETSI cannot guarantee their long term validity.

The following referenced documents are not necessary for the application of the present document but they assist the user with regard to a particular subject area.

The following referenced documents are not necessary for the application of the present document but they assist the user with regard to a particular subject area.

[i.1] Computers in Industry, Volume 134 (2022): "Implementation of digital twins in the process industry: A systematic literature review of enablers and barriers", Matteo Perno, Lars Hvam, Anders Haug, 103558, ISSN 0166-3615.

[i.2] ETSI TR 103 844: "SmartM2M; Digital Twins and Standardization Opportunities in ETSI".

[i.3] ETSI TS 103 845: "SmartM2M; Digital Twins Communication Requirements".

[i.4] IEEE™ Volume 108 Issue 10 (October 2020): "Digital Twin in the IoT Context: A Survey on Technical Features, Scenarios, and Architectural Models," R. Minerva, G. M. Lee and N. Crespi, pp. 1785-1824, doi: 10.1109/JPROC.2020.2998530.

[i.5] Alessandro Ricci, Angelo Croatti, Stefano Mariani, Sara Montagna, and Marco Picone: "Web of Digital Twins", ACM Trans. Internet Technol. 22, 4, Article 101 (November 2022), 30 pages.

[i.6] Journal of Manufacturing Systems, Volume 58, Part A, (2021): "Cyber-physical systems architectures for industrial internet of things applications in Industry 4.0: A literature review", Diego G.S. Pivoto, Luiz F.F. de Almeida, Rodrigo da Rosa Righi, Joel J.P.C. Rodrigues, Alexandre Baratella Lugli, Antonio M. Alberti, Pages 176-192, ISSN 0278-6125.

[i.7] W3C® - Web of Things - Documentation.

[i.8] ACM Trans. Internet Things 4, 1, Article 8 (February 2023): "A Flexible and Modular Architecture for Edge Digital Twin: Implementation and Evaluation", Marco Picone, Marco Mamei, and Franco Zambonelli, 32 pages.

[i.9] ETSI - The Smart Applications REFerence Ontology, and extensions (SAREF).

[i.10] IEEE™ Access, vol. 8 (2020): "Digital Twin: Enabling Technologies, Challenges and Open Research", A. Fuller, Z. Fan, C. Day and C. Barlow, pp. 108952-108971.

[i.11] ETSI STF 641: "SAREF Digital Twins", Technical Body: SmartM2M - Project No: 641.

[i.12] oneM2M: "Global IoT Technical Specifications".

3 Definition of terms, symbols and abbreviations

3.1 Terms

For the purposes of the present document, the following terms apply:

Digital Twin (DT): comprehensive software representation of an individual Physical Object

NOTE 1: Denoted also a Physical Twin.

NOTE 2: It includes the properties, conditions, relationships, events and behaviour(s) of the real-life object through models and data. A Digital Twin is a set of realistic models that can digitalize and simulate an object's behaviour in the deployed environment. The Digital Twin represents and reflects its physical twin and remains its virtual counterpart across the object's entire lifecycle [i.1].

3.2 Symbols

Void.

3.3 Abbreviations

For the purposes of the present document, the following abbreviations apply:

AF	Augmentation Function
API	Application Programming Interface
CoAP	Constrained Application Protocol
CPS	Cyber-Physical System
CPU	Central Processing Unit
DA	Digital Adapter
DI	Digital Interface
DT	Digital Twin
DTD	Digital Twin Description
DTM	Digital Twin Model
DTR	Digital Twin Representation
DTS	Digital Twin State
GPS	Global Positioning System

HTTP	Hypertext Transfer Protocol
HVAC	Heating, Ventilation and Air Conditioning
IIoT	Industrial Internet of Things
IoT	Internet of Things
JSON	JavaScript Object Notation
JSON-LD	JavaScript Object Notation for Linked Data
KPI	Key Performance Indicator
ML	Machine Learning
MNI	Management Interface
MOI	Monitoring Interface
MQTT	Message Queuing Telemetry Transport
PA	Physical Adapter
P-DTS	Pending Augmentation Function
PI	Physical Interface
PT	Physical Twin
PTD	Physical Twin Description
SAREF	Smart Appliances REFerence ontology
SLA	Service Level Agreement
TR	Technical Report
TS	Technical Specification
XML	eXtensible Markup Language

4 Digital Twin Abstraction & Core Functionalities

4.0 Foreword

As previously analysed and presented in ETSI TR 103 844 [i.2] and ETSI TS 103 845 [i.3] DTs stand as dynamic entities at the intersection of the physical and digital domains, encapsulating a spectrum of core functionalities that collectively define their essence and purpose. This clause delves into the profound core functionalities that define the very essence and purpose of DTs, shedding light on their interactions with the tangible and intangible worlds, their adept management of internal models, and their journey through the intricate life cycle.

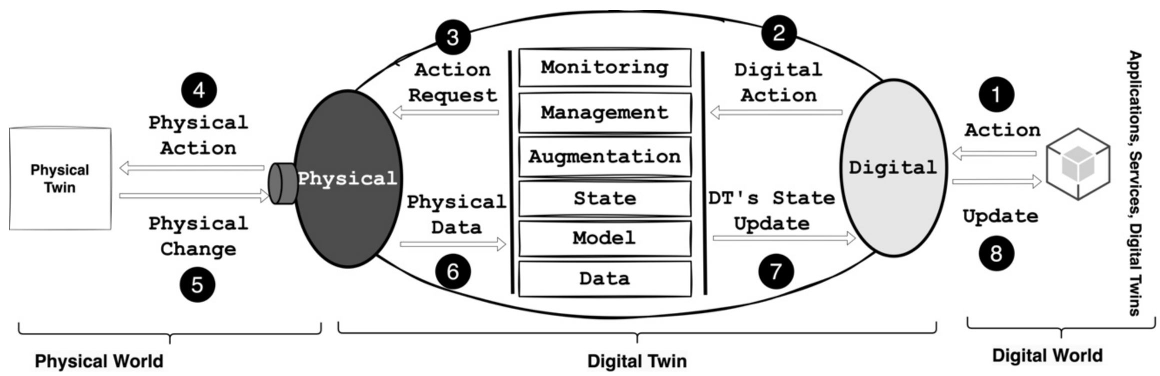

Figure 1: High Level representation of cyber-physical relationships and communication through Digital Twins

A fundamental architectural focus lies on the effective identification of the DT characteristics in terms of interaction flows and communication specifications in the design, implementation, and deployment of standardized and interoperable DTs. The effective functioning of DTs in today's interconnected world heavily relies both on their seamless communication across various communication components and interaction flows taking into account both the physical and the digital layers together with a structed architecture able to shape the DT behaviour together with this life-cycle, to support its deployment and monitoring and to enable a flexible manageability in distributed ecosystems as schematically depicted in the building blocks of Figure 1.

DT's communication layers and components, spanning both the physical and digital dimensions of a DT's core, form the foundation for robust interactions and collaborations, ultimately enabling the vision of an ecosystem of interconnected DTs sharing common architectural principles but that can be implemented and deployed across different runtimes and location through the edge-to-cloud compute continuum. By emphasizing the significance of structured communication channels, this clause recalls the main the features and prerequisites crucial for designing and building an effective DT architecture achieving interoperability, ensuring that DTs can coexist harmoniously, exchange critical data, and contribute to the broader operational landscape.

At its core, a DT is a sophisticated entity with several key responsibilities and modules that collectively enable its multifaceted functions. The *Data* module shall serve as the foundational component, tasked with handling, distributing, collecting, processing, and storing information derived from the mirrored physical asset. It shall also maintain the DT's current state and retains a historical record of past states, encompassing real-world data such as sensor readings, equipment status, and environmental conditions.

The *Model* component shall shape and embody the essence of a DT, incorporating intricate logic to mirror and digitalize the target physical twins into precise digital replicas. It should maintain a close relationship with the Data module, leveraging real-world data to craft a detailed digital representation of the physical asset. Moreover, the Model module shall be in charge of computing the new DT's state based on information gathered from the physical world and the DT's design, encompassing the physical asset's structure, behaviour, and attributes.

The *State* module plays a pivotal role and shall oversee structuring the current representation of the DT as a synchronized representation of its physical counterpart. It should interact with the Model for the computation and shall capture the status and condition of the twin as a reflection of the interconnected physical world, tracking changes and updates to ensure accurate digital representation. Additionally, the State module structures and preserves a comprehensive list of properties, events, relationships, and actions available on the twin over time, enhancing interoperability by providing a common framework for multiple DTs to share.

Introducing an architectural module focused on *Augmentation* brings adaptability to the DT, shall allow the enhancement, modification, or extension of its capabilities over time, and should work side by side with the Model and the State in order to extend the nature and the capabilities of the target DT. This adaptability includes adding new properties, relationships, behaviours, or components to keep the DT synchronized with evolving needs.

The *Management* components should be responsible for overseeing the DT's lifecycle, including creation, deployment, operation, and retirement. They ensure seamless configuration and integration into the operational context while managing interactions with other systems and services.

Lastly, the *Monitoring* module should be responsible of observing and evaluating the DT's performance, health, and behaviour, focusing on the quality of the cyber-physical relationship between the DT and its associated physical twins. It identifies issues, anomalies, and optimization opportunities, providing valuable insights into the DT's operation and its impact on the physical asset.

These core components collectively form an advanced architectural goal that constitutes a DT, bridging the physical and digital dimensions, empowering decision-making processes, and delivering invaluable insights across various applications and industries. These envisioned modules play pivotal roles in physical and digital communications, facilitating the realization of the DT's cyber-physical capabilities. Subsequent clauses delve deeper into characteristics and specifications associated with fundamental modules, physical and digital communications of a DT, its description, and discoverability, shedding light on their interactions with these core components.

4.1 Physical Communication

Interacting with the physical world in the realm of the Internet of Things (IoT) presents a multifaceted landscape rife with challenges and complexities that should be navigated in the creation and definition of a Digital Twin. One of the foremost challenges stems from the diverse array of devices and systems comprising the IoT ecosystem. These devices range from simple sensors gathering environmental data to sophisticated industrial machinery orchestrating complex manufacturing processes. Each device often operates on its unique communication protocols, data formats, and interfaces, complicating efforts to integrate them into a cohesive digital framework. For instance, a smart city deployment may involve sensors measuring temperature, humidity, and air quality, each employing different communication protocols such as Zigbee®, LoRaWAN, or Wi-Fi® or application layer solutions such as MQTT, HTTP and CoAP. Harmonizing these disparate technologies to create a unified digital representation requires careful orchestration and standardization efforts.

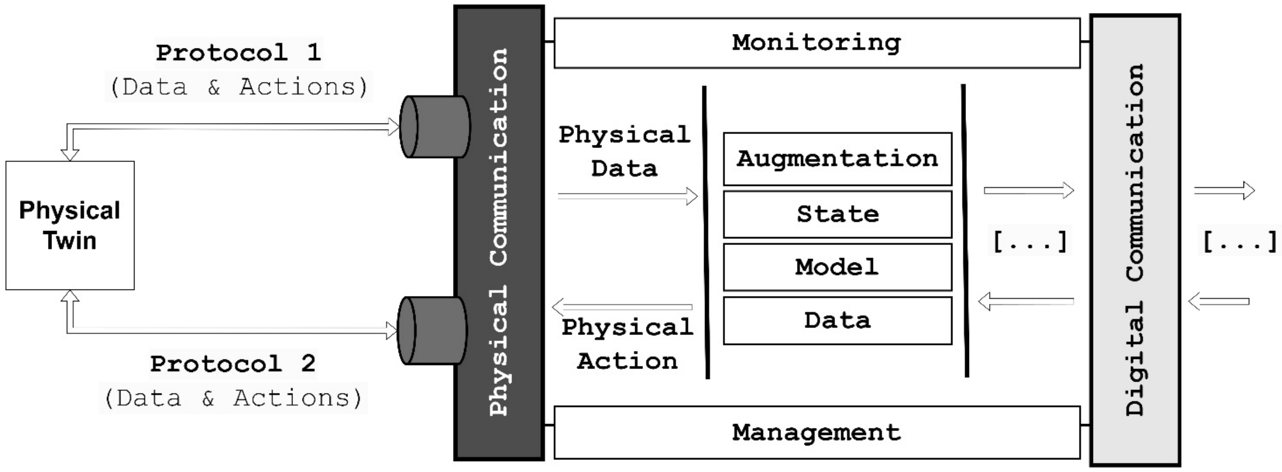

Figure 2: Physical Interaction between the DT and the Physical World

Real-time data acquisition and synchronization pose another significant challenge in the context of IoT-driven DTs. In environments characterized by rapid changes and dynamic conditions, ensuring that the DT accurately mirrors the state of its physical counterpart becomes paramount. Consider a smart grid system tasked with managing electricity distribution across a city. In such a scenario, the digital twin should continuously receive updates on power consumption, grid stability, and potential outages to make informed decisions in real-time. Achieving this level of synchronization demands robust communication infrastructures, low-latency data processing capabilities, and efficient algorithms for state reconciliation.

Despite these challenges, the creation of DTs holds immense promise for revolutionizing various industries and domains. From optimizing supply chains and streamlining operations to enabling predictive maintenance and personalized healthcare, DTs offer unparalleled insights and capabilities. By addressing the challenges associated with interacting with the physical world in the IoT context, organizations can harness the full potential of DTs to drive innovation, efficiency, and sustainability in the digital age.

This fragmentation poses a direct impact on the physical communication of a DT since it interacts with physical assets and sensors distributed across various IoT networks, the ability to discover and access these resources becomes paramount for its operation. DT shall include a structured and modular way to handle interactions and protocols management with Physical Twins (PTs) trough different protocols and technologies as illustrated in Figure 2 in order to avoid that a DT may struggle to establish connections with its physical counterparts, leading to communication breakdowns, data inconsistencies, and operational inefficiencies. Interactions and communication between the DT and the associated PT shall be bidirectional (if supported by the PT) allowing at the same time both the possibility of reading variation of PT's state and the transmission of action to the physical counterpart to change status, trigger actions or change behaviour.

Furthermore, the physical communication side of a DT architecture should oversee decoupling the complexity of the physical world from the core of the DT that should be potentially reusable across different DT instance associated to the same class of PTs that are using different communication protocols or data format.

EXAMPLE 1: In a smart building scenario or a generic Cyber-Physical System (CPS) related to an industrial use case [i.6] where the DT monitors HVAC systems, lighting, and occupancy sensors, fragmented protocols could impede its ability to retrieve real-time data or issue control commands, compromising its effectiveness in optimizing building performance and energy efficiency.

Breaking down the physical communication channel of a DT into modular parts is crucial. This approach brings many benefits, such as improving flexibility, scalability, and maintainability. When the physical communication channel is organized into modules, each one handling different tasks or protocols, DT can adjust better to different needs and situations. DT's architecture should enable modularity for the interaction and communication with PTs allowing the reuse of different components across multiple twin instances, the separation of concerns in the development and deployment together with the simplicity of adopting different protocols, communication patterns and data formats at the same time without interferences among active entities.

One of the main issues when it comes to DTs and their connection to the physical world, especially in the realm of IoT, is understanding how they can know what's in their physical counterpart, the physical twin. In IoT, there are many different techniques for finding and using resources and services, and this makes it difficult for DTs to understand what is available on the asset to digitalize in terms of properties, events, actions, and relationships. As schematically depicted in Figure 3, a DT architecture shall support and enable DTs to be able to discover this information, and this can happen interacting directly through IoT protocols that natively support asset description denoted also as Physical Twin Description (PTD) and discoverability (e.g. CoAP), or through specific configurations provided to the DT to help it find the information it needs (e.g. structuring the MQTT topics useful to find the information required on the PT). This information includes things like what properties and actions the PT has, and how the digital twin can access them. Essentially, the description of the PT is a set of important information that shall help the DT to start doing its digitalization process. Also in this case, the Physical Communication layer of a DT architecture shall be in charge of decouple this discovery process from the core of the DT allowing it to be aware of what is available on the physical side without expose the complexity of retrieving the required information.

EXAMPLE 2: Please consider a DT of a smart thermostat that's meant to control the temperature in a building. In order for the DT to work effectively, it needs to know things like the current temperature in the building, whether the heating or cooling systems are on or off, what the desired temperature settings are, and the protocol adopted to retrieve that information and the data format that represent them. This information should be part of the description of the PT, and the DT would use it to receive telemetry data, make decisions and take actions to maintain the desired temperature interacting in a bidirectional way with the physical world.

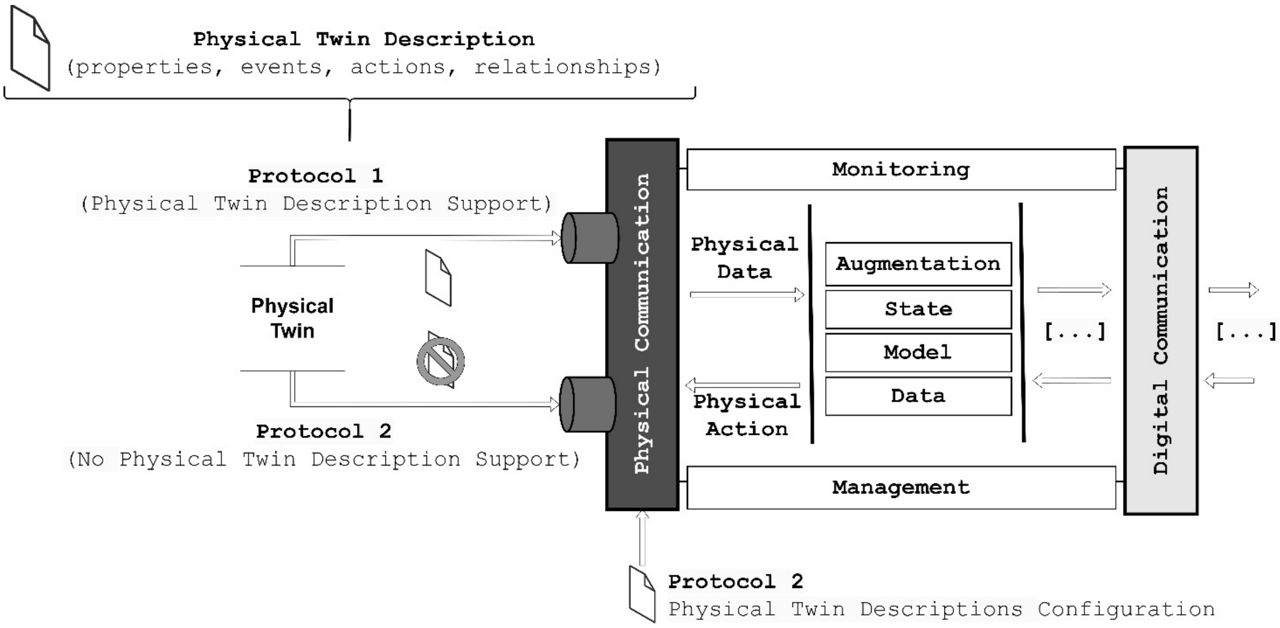

Figure 3: Decoupling Physical Interaction from the DT Core through a Physical Twin/Asset Description

In the IoT landscape, the integration of DTs with the physical layer presents a complex challenge, particularly concerning the decoupling of DT core functionalities from the intricacies of physical communication protocols. Once physical capabilities and resources are discovered, the DT's core shall be independent from this complexity. This necessitates that the physical interface, or communication channel, of the DT conceals the specifics of protocols and instead focuses solely on describing the physical capabilities of the associated objects. This abstraction is crucial for ensuring interoperability and scalability, especially when a DT is responsible for digitizing multiple physical assets. Regardless of the underlying communication protocols used by these assets, the DT's core should rely on a high-level description that remains consistent across diverse devices. For instance, an object with identical capabilities may communicate through different protocols, yet the digital twin's description of its physical assets remains uniform. In essence, the physical communication layer of the DT should act as a barrier, shielding the core from the complexities of physical communication, while leveraging standardized descriptions to facilitate digitalization protocols and interaction with the physical world.

4.2 Digital Communication

Interacting with the digital world presents several challenges and complexities in the creation and definition of a DT architecture, especially concerning their representation for seamless interoperability within the digital ecosystem. A key challenge arises from the heterogeneous nature of digital systems, each employing disparate data formats, communication protocols, and platforms.

For instance, consider a scenario where a DT of an industrial machine needs to communicate with various digital applications, such as maintenance monitoring systems, supply chain management platforms, and predictive analytics tools. Each of these applications may use different data formats and protocols, making it challenging for the digital twin to exchange information effectively. In this context, it is crucial to establish standardized data formats, communication protocols, and security mechanisms to facilitate seamless interoperability and secure data exchange. Additionally, representing digital twins in an interoperable manner across different platforms and applications requires standardized descriptions and interfaces.

EXAMPLE 1: Adopting industry-standard ontologies or data models, such as Smart Applications REFerence Ontologies (SAREF) [i.9] and specific extension for DTs [i.11], can enable consistent representation and interaction of DTs across various domains and applications. By adhering to established standards and protocols, twins can effectively integrate with diverse digital systems, enabling enhanced collaboration, data exchange, and decision-making capabilities within the digital ecosystem.

Modularity plays a critical role in the design of DT architecture and in particular in the creation and definition of the digital communication channel of a DT, offering several key advantages that enhance flexibility, scalability, and maintainability. A DT architecture should structure the communication channel into modular components, each responsible for specific functionalities or protocols, DTs should be able to adapt to varying requirements and environments more effectively. For instance, consider a DT operating in an IoT environment where it needs to interact with multiple devices using different communication protocols such as MQTT, CoAP, and HTTP. By modularizing the communication channel, the DT architecture can incorporate separate adapters for each protocol, allowing it to seamlessly communicate with diverse devices without the need for extensive modifications to its core functionality.

Moreover, modularity facilitates scalability by enabling easy integration of new communication protocols or functionalities as the system evolves.

EXAMPLE 2: If a new IoT device is introduced into the ecosystem using a different communication protocol, the DT should be able to add in a simplified way a new module without disrupting its existing operations and interfering with other digital communication and interactions already in place. Additionally, modularity enhances maintainability by isolating components, making it easier to troubleshoot and debug issues within the communication channel. If a particular protocol adapter encounters a problem, it can be addressed independently without affecting other components, reducing downtime and improving system reliability.

Furthermore, modularity promotes interoperability by allowing DTs to communicate with a wide range of external systems and services. Each modular component should be designed to adhere to standardized protocols and interfaces, facilitating seamless integration with third-party applications and platforms. In essence, modularity on the digital communication channel of a DT enables flexibility, scalability, maintainability, and interoperability, making it a fundamental aspect of its design and implementation.

DT architectures shall have access to a structured DT state accessible through the Digital Communication with the aim to provide a uniform, interoperable and potentially standardized representation of the DT's capabilities, enabling a simplified integration with external systems, applications, and services. By exposing its state in a structured format, the DT can communicate its status, properties, and functionalities in a universally understandable manner, regardless of the underlying communication protocols or technologies and supporting at the same time multiple protocols and communication technologies.

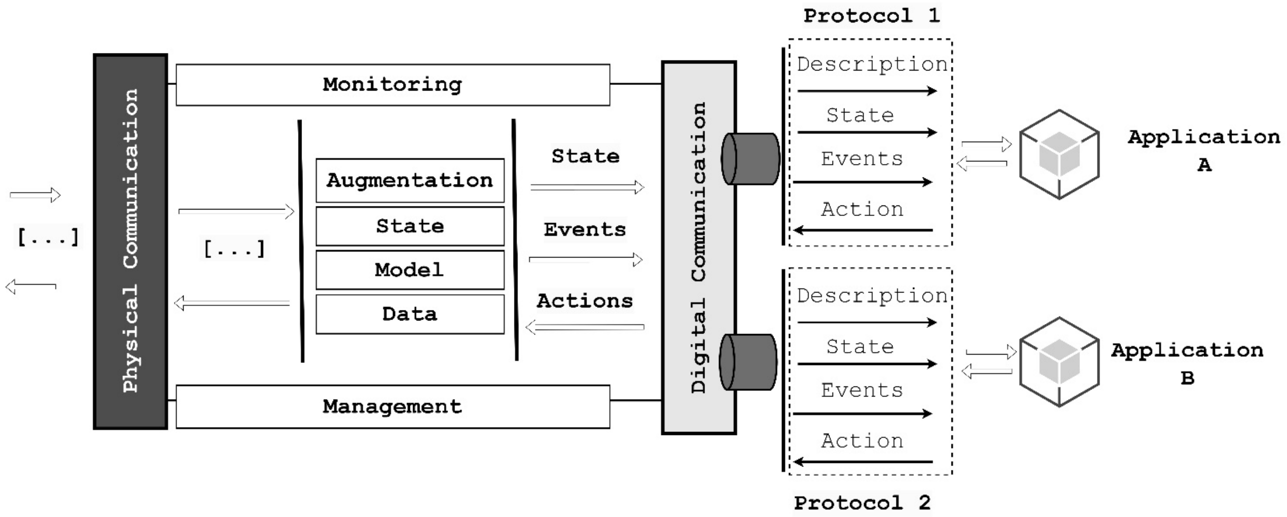

Figure 4: Representation of the DT in the Digital Space through its description, current state and events together with the support of actions

Moreover, a well-defined DT state promotes interoperability by adhering to established data formats and standards. For instance, using formats like JSON or XML ensures compatibility with a wide range of digital systems and applications. This interoperability enables the DT to interact seamlessly with various platforms, including IoT networks, cloud services, and enterprise software, facilitating data exchange and collaboration across diverse environments.

As illustrated in Figure 4 a DT architecture shall expose the current state of the DT as the fundamental functionality of a twin in charge of digitalizing a PT and providing its digital replica and representation in the cyber space. Additionally, a structured DT state enhances the scalability and flexibility of the Digital Twin ecosystem. As new devices, sensors, or functionalities are added to the DT, they can be easily incorporated into its state representation without requiring significant modifications to the communication layer. This scalability ensures that the DT remains adaptable to evolving requirements and can accommodate changes in its environment or operational context. Furthermore, a structured DT state supports effective monitoring, control, and management of the DT instance. External systems can query the DT state to retrieve real-time information about its status, retrieve and analyse historical data, or trigger specific actions based on predefined conditions. This capability enables proactive decision-making, predictive maintenance, and optimization of DT performance, leading to improved operational efficiency and performance.

Together with the state, a DT architecture should also expose on the digital side the events generated by the twin and associated for example to variation of its context and detected anomalies. The declaration of the potential generated events shall be in the state description as a part of the nature and structure of the DT itself while the generation should be communicated through different interaction pattern.

A description of the DT instance should be provided to describe the nature of the DT, active internal components, and contextual information such as the current software version or the runtime on which the twin is executed, and it is running. An additional core aspect that shall be adopted by a DT architecture in its digital communication capabilities is the possibility to receive digital actions to be managed by the twin and potentially forwarded to the physical world to trigger specific behaviour and actuate the environment.

In summary, having a structured and interoperable DT state accessible through the Digital Communication layer is essential for facilitating seamless integration, promoting interoperability, enhancing scalability and flexibility, and enabling effective monitoring and management of DT instances. By providing a standardized representation of capabilities, the DT state shall serve as a foundation for building interconnected and intelligent systems that can thrive in today's complex digital ecosystems.

4.3 Digital Twin Model

The model of a DT shall serve as the foundational element responsible for defining and executing the behaviour of the twin, dictating how it interacts with its physical counterpart and presents its state to the digital world. In essence, the model shall be the driving force behind the digitalization process. It shall play the pivotal role in handling events and data originating from the physical entity, leveraging this information the model shall compute and maintain updated the state of the DT in terms of its properties, events, relationships and enabled actions and behaviours. Simultaneously, the model shall manage incoming digital actions received from the digital world, ensuring their feasibility, and determining whether they should be translated into actions for the physical entity. Essentially, the model shall orchestrate the digitalization process while also facilitating the implementation of augmentation functions for the DT. These augmentation functions serve to enhance the capabilities of the twin, whether by updating its state with newly computed properties or generating events for the digital communication channel. Thus, the model should operate through three primary phases: *digitalization*, *augmentation*, and *actionability*. These functionalities should integral components of the digital model within a DT architecture, where the software framework should enable the seamless definition and implementation of these model concepts as schematically illustrated in Figure 5.

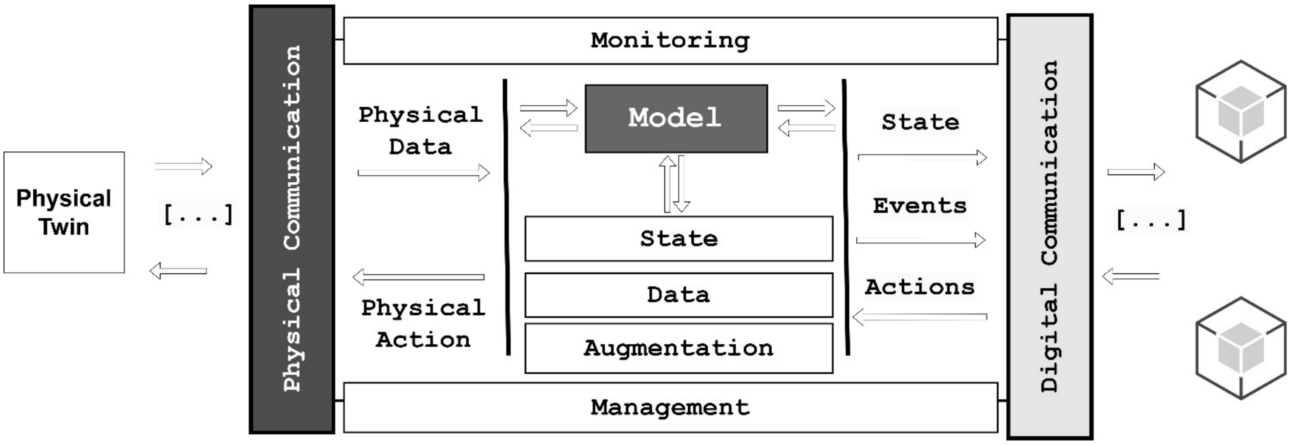

Figure 5: High-level representation of the DT Model and its relationships with both the Physical and the Digital side of the DT together with the internal modules

The DT Model should utilize the description of the physical assets as a blueprint to comprehend the intricacies of the physical world and determine the appropriate course of action for digitalization. By leveraging the information provided in those description, such as the properties, relationships, and events associated with the physical asset, the DT Model should gain a comprehensive understanding of the available resources and capabilities. This understanding enables the model to make informed decisions on how to handle the digitalization process effectively.

EXAMPLE 1: If the physical description coming from the physical communication channel indicates that a sensor is capable of measuring temperature and humidity, the DT Model can infer the types of data to expect and how to interpret them. Similarly, if the description highlights relationships between different physical assets, such as a sensor's connection to a control system, the DT Model can ensure that these relationships are accurately represented in the digital twin. Overall, the DT Model should rely on the information and description coming from the physical world through the physical communication channel as a crucial source of information to guide its digitalization strategies and ensure a faithful representation of the physical world in the digital realm.

EXAMPLE 2: Consider a DT representing a smart thermostat in a building automation system. The model of this DT would be responsible for capturing data from sensors measuring temperature, humidity, and occupancy within the building. Based on this data, the model computes the current state of the thermostat, such as the desired temperature setting or HVAC mode. Additionally, the model may receive digital commands from a building management system to adjust the thermostat's settings based on occupancy schedules or energy-saving goals. If the received command aligns with the constraints and capabilities defined in the model, it is translated into a physical action, such as adjusting the HVAC system's parameters. Furthermore, the model can include augmentation functions to optimize energy efficiency or predictive maintenance, leveraging historical data and machine learning algorithms to enhance the DT's capabilities over time.

4.4 Digital Twin State

In essence, DTs aim to digitalize and represent their physical counterparts, yet capturing every detail may not always be practical or necessary. A DT should be modelled with specific goals in mind and tailored to its operational context. It should at least encompass essential properties, events, behaviours, and relationships relevant to its function, such as optimizing energy usage in a building or detecting anomalies of an industrial machine.

In certain scenarios, not all attributes of the physical object are pertinent to the DT's purpose. In such cases, irrelevant attributes are omitted from the DT representation and description. This contextualization ensures that only pertinent features and data are included to accurately represent the physical object in the virtual space under consideration.

Following these principles, the foundational goal of a DT should be the capability to dynamically build and maintain a comprehensive digitalized representation of the state of its physical counterpart through the definition and computation of the DT state by encapsulating describing properties, events, actions, and relationships.

The State of a DT (DTS) embodies various critical elements starting from the fact that each DT shall have a unique global identifier for identification purposes and shall have a reference to the ids or the PTs that it is representing and digitalizing. The DT's state shall convey information about associated physical counterparts, with identifiers tailored to specific application domains, supplemented by metadata for external application comprehension.

Properties within the DT's State should encompass observable attributes of the corresponding PT, dynamically evolving to reflect real-world changes and providing a foundational layer for constructing detailed snapshots. Events at the PT domain level should contribute to the DTS by recording changes and transitions, offering temporal dimension and historical context to the DT's representation.

Actions within the DTS should extend beyond observation, enabling bidirectional interaction between digital and physical domains, dynamically contributing to the DT State and maintaining synchronization.

Relationships in the DTS should enable to establish connections with other DTs, reflecting real-world dependencies and associations, enhancing contextual understanding through dynamic and adaptable links.

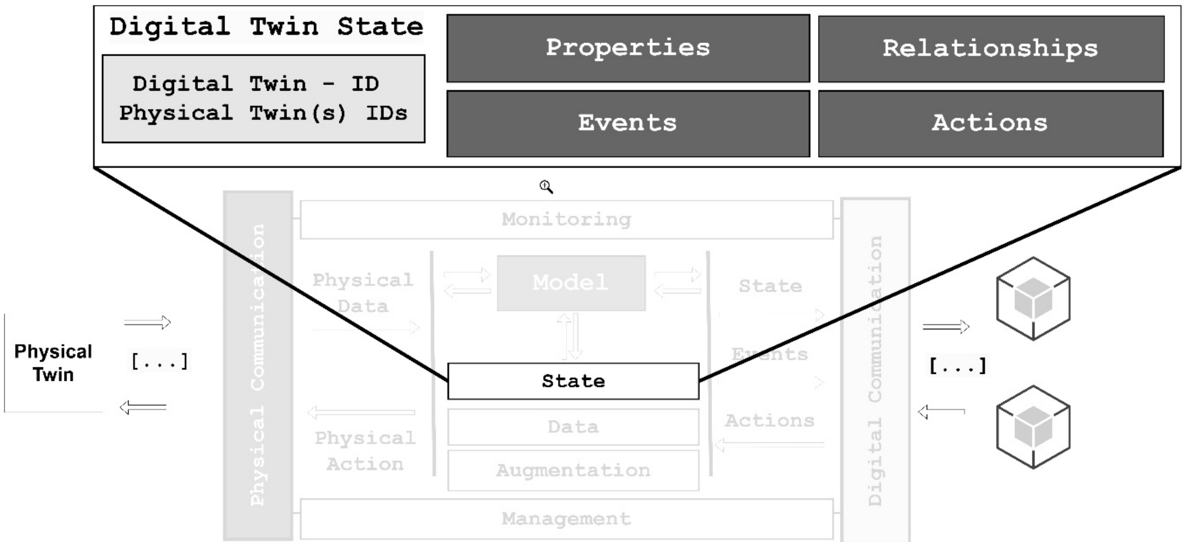

Figure 6: Schematic visual representation of the DT State with its field and characteristics

The synergy of properties, events, actions, and relationships within the DTS should support an active and evolving model of the PT. As properties dynamically shift, events unfold, actions are executed, and relationships adapt, the DT's State provides a synchronized reflection and representation of the characteristics of in the physical world and its evolution overtime.

As depicted in Figure 6, within a DT software architecture, the DTS operates alongside the DT's model. The model, exemplified by a smart home automation system, computes the new state based on user-defined behaviours and adjustments from the physical environment received through the DT's physical communication channel. For instance, if the user sets the temperature to 25 °C and the sensor detects a room temperature of 22 °C, the model calculates new state and the appropriate action to adjust the thermostat.

Both the model and state remain decoupled from the complexities and fragmentation of the physical layer, facilitated by the mediation provided by the DT's physical layer.

EXAMPLE 1: In a smart city infrastructure with various interconnected IoT devices, the DT's physical layer ensures seamless communication and data exchange between devices, such as streetlights, sensors, and traffic signals.

On the other hand, the DTS serves as the primary information source within each DT instance. In the context of an industrial IoT system, the DTS continuously feeds data to the digital communication channel of the DT, presenting the current state through multiple protocols and communication channels. This allows for real-time monitoring of equipment performance, energy usage, and production metrics, enhancing operational efficiency.

Furthermore, the DTS and its computation support DT history and storage, enabling the tracing of the twin's evolution over time.

EXAMPLE 2: In an agricultural IoT application, the DTS stores historical data on soil moisture levels, temperature, and crop growth, facilitating data-driven decision-making and predictive analytics. Additionally, augmentation functions can extend the state and capabilities of a DT, such as implementing machine learning algorithms to optimize energy usage in a smart building. However, the responsibility for digitizing and synchronizing the physical and digital states remains with the model, ensuring consistency and accuracy in the representation of the physical world.

4.5 Digitalization Process

The Digitalization Process of a DT, also known as reflection or shadowing, involves multiple steps to ensure an accurate representation of the physical asset's state in the digital realm and enable bidirectional communication between the physical and digital worlds through the DT Model.

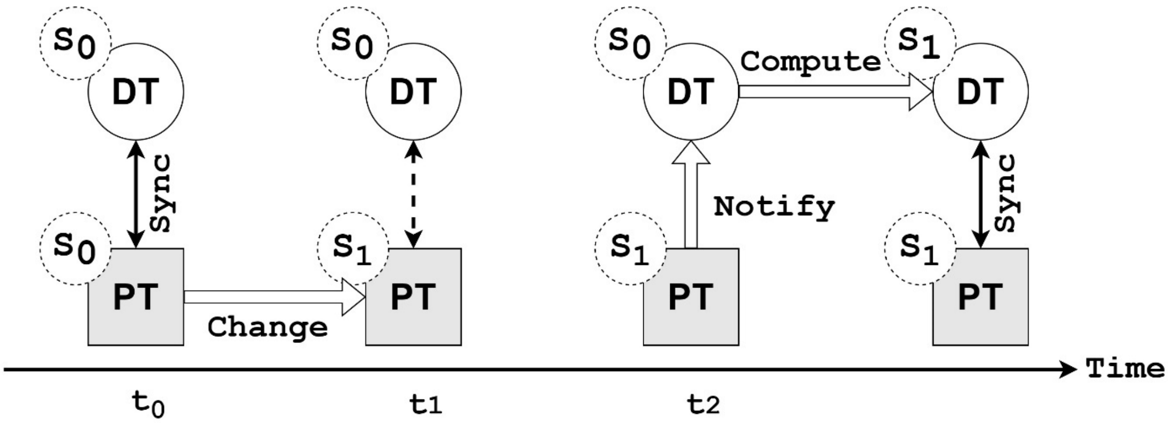

Figure 7: Involved steps associated to the cyber-physical digitalization process of a DT from Physical to Digital

The process of updating the state of a DT from its physical counterpart involves several steps that should be considered in the design and implementation of a DT Software Architecture. Firstly, any relevant changes in the state of the physical asset are captured.

EXAMPLE 1: In a smart factory setting, sensor readings may detect a decrease in machine temperature or an increase in production output.

Next, these captured events are transmitted to the Digital Twin Model, where they are interpreted and processed. The Model relies on the information provided in the Physical Asset Description to understand the significance of these events and their implications for the Digital Twin.

Finally, based on the information received from the physical world and the logic defined in the DT Model, the state of the Digital Twin is updated accordingly. This update could involve modifying existing properties, creating new ones, or triggering events within the Digital Twin. For instance, in the case of a smart home system, a change in temperature detected by a thermostat sensor could prompt the Digital Twin to adjust the HVAC settings accordingly.

With respect to the sequence illustrated in Figure 7 it is possible to identify the following steps with the associated responsibilities:

- **Flow 1 - From Physical to Digital:**
    - **T_0 - Synchronized Start:** At the beginning both PT and DT are synchronized, and their states are aligned according to the current DT's model and previous interactions and data exchange.
    - **T_1 - Capture Physical State Changes:** The process begins with capturing any relevant changes in the state of the physical asset. This includes detecting events such as sensor readings, equipment status changes, or environmental fluctuations. At this step, the State of the PT moved from $\mathbf{S_0}$ to $\mathbf{S_1}$ but the DT state is still at the previous State $\mathbf{S_0}$.
    - **T_2 - Propagation to DT Model:** The captured events are then propagated to the DT Model, where they are interpreted and processed. The DT Model uses the information provided in the PT Description to understand the significance of these events and determine their impact on the digital twin's state. At this step, the DT has been notified through is physical communication channel about PT's variation an starts its digitalization process to compute the new State but states are still not aligned with an $\mathbf{S_1}$ on the PT and $\mathbf{S_0}$ on the DT.
    - **T_3 - Update DT State:** Based on the information received from the physical world and the logic defined in the DT Model, the state of the DT is updated accordingly. This update may involve modifying existing properties, creating new ones, or triggering events within the DT. Finally, at this step the two States of PT and DT are synchronized and aligned according to the DT's Model at $\mathbf{S_1}$.

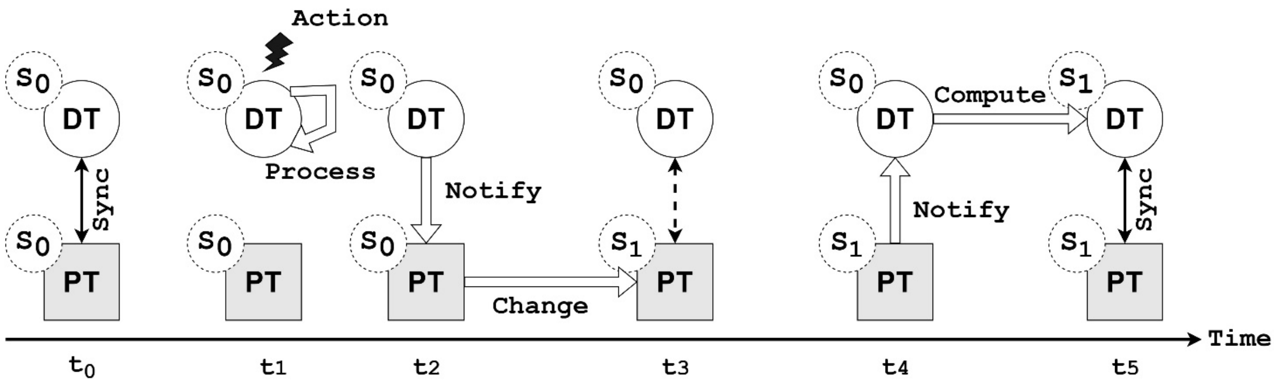

Figure 8: Involved steps associated to the cyber-physical digitalization process of a DT from Digital to Physical

The interaction and digitalization between the PT and the DT encompass not only the flow of information from the physical to the digital realm but also extend to actions originating from the digital world. For instance, applications may trigger actions within the DT that require validation and subsequent forwarding to the PT for execution. Any variations occurring within the PT because of these actions are then reflected in the DT, following the steps previously described in Flow 1. This bidirectional interaction ensures that the DT remains synchronized with its physical counterpart, facilitating real-time monitoring, control, and decision-making across both domains.

In Flow 2, the interaction from the digital to the physical domain unfolds as follows. At the beginning, both the PT and DT commence in synchronized states, aligned according to the current DT's model and prior interactions. After that, the DT receives a request for action from the digital world. For instance, a smart home application may command the DT to adjust the thermostat temperature. This action undergoes validation by the DT Model to ensure compatibility with the physical asset's capabilities and constraints, as outlined in the PT Description.

Upon validation, the action is transmitted from the DT to the physical asset. This process entails converting the digital command into a format understandable by the physical system and transmitting it through the appropriate communication channel. Subsequently, the physical asset executes the requested action, potentially resulting in changes to its state, such as adjusting parameters or activating components.

After that, the captured events are relayed to the DT Model (DTM) for interpretation and processing. This information, derived from the PT Description, guides the DT in understanding the implications of the physical asset's changes on its state. Following this, the DT's state is updated based on the received information and the logic defined in the DT Model. This update aligns the DT's state with that of the PT, ensuring synchronization according to the DT's model. For instance, if the smart home thermostat adjusts the temperature, the DT's state would be updated to reflect this change, ensuring alignment between the digital and physical domains.

With respect to this second flow and the sequence illustrated in Figure 8 it is possible to identify the following steps with the associated responsibilities:

- **Flow 2 - From Digital to Physical:**
  - **T_0 - Synchronized Start:** At the beginning both PT and DT are synchronized, and their states are aligned according to the current DT's model and previous interactions and data exchange.
  - **T_1 - Request Digital Action:** In this step, the DT receives a request for action from the digital world. This could be a command to perform a specific task, adjust a parameter, or initiate a process. The received action should be validated by the DT Model to ensure its feasibility and compatibility with the physical asset. The model checks whether the requested action aligns with the capabilities and constraints of the physical asset as described in the PT Description.
  - **T_2 - Propagation to Physical Asset:** Once validated, the action is propagated from the DT to the physical asset. This may involve converting the digital command into a format understandable by the physical system and transmitting it through the appropriate communication channel.
  - **T_3 - Execution by Physical Asset:** Finally, the physical asset executes the requested action based on the received command. This action could result in changes to the physical state, such as adjusting operating parameters, activating or deactivating components, or initiating a process. At this step, the State of the PT moved from $\mathbf{S_0}$ to $\mathbf{S_1}$ but the DT state is still at the previous State $\mathbf{S_0}$.
  - **T_4 - Propagation to DT Model:** The captured events are then propagated to the DT Model, where they are interpreted and processed. The DT Model uses the information provided in the PT Description to understand the significance of these events and determine their impact on the digital twin's state. At this step, the DT has been notified through is physical communication channel about PT's variation an starts its digitalization process to compute the new State but states are still not aligned with an $\mathbf{S_1}$ on the PT and $\mathbf{S_0}$ on the DT.
  - **T_5 - Update DT State:** Based on the information received from the physical world and the logic defined in the DT Model, the state of the DT is updated accordingly. This update may involve modifying existing properties, creating new ones, or triggering events within the DT. Finally, at this step the two States of PT and DT are synchronized and aligned according to the DT's Model at $\mathbf{S_1}$.

Throughout the Digitalization Process, the DT Model should serve as a crucial intermediary, facilitating seamless communication and synchronization between the physical and digital realms. By accurately capturing and processing information from the physical world and translating digital commands into physical actions, the digital twin enables effective monitoring, control, and optimization of physical assets in real-time.

Summarizing, considering a running example associated to a smart building scenario where a DT is employed to monitor and manage the Heating, Ventilation, and Air Conditioning (HVAC) system. Here's how the DT Digitalization Process might unfold in this context:

- **Flow 1 - From Physical to Digital:**
  - **Capture Physical State Changes:** Sensors installed throughout the building continuously monitor temperature, humidity levels, and occupancy in different rooms. These sensors detect changes in the environment, such as a room becoming too warm due to sunlight or overcrowded due to a meeting.
  - **Propagation to DT Model:** The sensor data is sent to the DT Model, which interprets the incoming information. The DT Model understands the significance of each data point based on the Physical Asset Description, which includes details about the HVAC system's components, sensor locations, and expected operating conditions.
  - **Update DT State:** Using the information from the sensors and the logic defined in the DT Model, the digital twin updates its state.

EXAMPLE 2: If the temperature in a particular room exceeds a predefined threshold, the DT might trigger an event to activate the HVAC system to cool the room down.

- **Flow 2 - From Digital to Physical:**
  - **Request Digital Action:** An occupant of the building uses a mobile app to adjust the temperature settings in their office. They request the DT to lower the temperature by a few degrees to make the room more comfortable.
  - **Validation by DT Model:** The DT Model receives the request and checks whether the proposed temperature adjustment is within the permissible range set for the HVAC system. It also verifies if there are any conflicting commands or safety concerns.
  - **Propagation to Physical Asset:** Once validated, the DT forwards the command to the HVAC system through the appropriate communication protocol, such as MQTT or CoAP. The command instructs the HVAC system to adjust the temperature settings accordingly.
  - **Execution by Physical Asset:** The HVAC system receives the command and executes the requested action, such as adjusting the airflow or activating the heating or cooling mechanisms. As a result, the temperature in the office changes to the desired level.
  - **Execution of Flow 1 Steps:** After the previous steps, the digitalization process restart from Flow1 since a variation of the physical world has been detected on PT and should be synchronized on the associated DT.

In this example, the DT Digitalization Process enables real-time monitoring of environmental conditions in the building and seamless interaction with the HVAC system to maintain occupant comfort and optimize energy efficiency. By leveraging sensor data and digital commands, the DT facilitates intelligent decision-making and proactive management of the physical environment, ultimately enhancing the overall building performance and user experience.

4.6 Digital Twin Augmentation

DT augmentation represents the process of enhancing the capabilities and functionality of a DT beyond its basic digital representation of the physical entity. This augmentation should be deeply intertwined with the DT Model, which serves as the blueprint defining the behaviour and characteristics of the DT. The augmentation process leverages the DT Model to extend the DT's capabilities, enabling it to perform advanced functions, adapt to changing conditions, and interact more intelligently with both the physical and digital worlds.

At its core, DT augmentation involves the integration of additional features, algorithms, or data processing mechanisms into the DT Model to enable new functionalities. These augmentations can take various forms, such as:

- **Advanced Analytics:** Integrating machine learning algorithms or predictive analytics models into the DT Model to enable the DT to forecast future states or detect anomalies in real-time.

EXAMPLE 1: In a manufacturing setting, a DT augmented with predictive maintenance capabilities can analyse historical data from equipment sensors to predict potential failures and schedule maintenance proactively, thereby minimizing downtime.

- **Dynamic Adaptation:** Implementing adaptive algorithms within the DT Model that allow the DT to dynamically adjust its behaviour based on changing environmental conditions or user requirements. For instance, a smart home DT augmented with adaptive lighting control can adjust the brightness levels of lights based on the time of day, occupancy patterns, or natural lighting conditions.
- **Integration with External Data Sources:** Enhancing the DT's ability to integrate and analyse data from external sources, such as weather forecasts, traffic patterns, or social media feeds. This augmentation enables the DT to contextualize its understanding of the physical environment and make more informed decisions.

EXAMPLE 2: A smart city DT augmented with real-time traffic data can optimize traffic signal timings to reduce congestion and improve traffic flow.

- **Interoperability:** Enhancing the DT's interoperability with other digital systems or devices through standard protocols and APIs. This augmentation enables seamless communication and collaboration between different DTs and digital services, facilitating data sharing and coordinated decision-making.

EXAMPLE 3: A supply chain DT augmented with interoperability features can exchange data with logistics partners' DTs to optimize inventory management and delivery schedules.

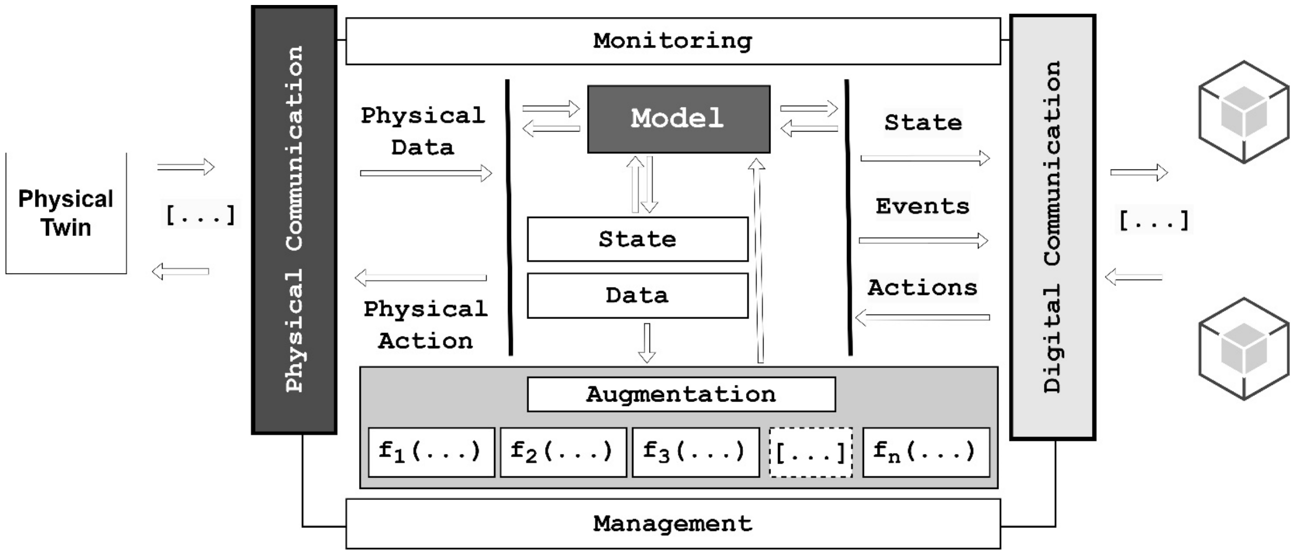

Figure 9: Augmentation Representation within a DT's Architecture with the interaction with both DT's Model, State and Data

As schematically represented in Figure 9, the augmentation of a DT can be achieved through various means, leveraging both the DT Model and the extension of interaction capabilities via physical and digital communication capabilities and channels. The DT Model serves as the foundation for implementing augmentation functionalities, allowing for the integration of advanced algorithms, analytics, and data processing mechanisms to enhance the DT's capabilities. Additionally, extending the DT's interaction capabilities through digital adapters enables the DT to go beyond the original capabilities of the physical entity.

EXAMPLE 4: By integrating additional sensors, actuators, or external data sources through digital adapters, the DT can gather more comprehensive data, perform advanced analytics, and interact with a broader range of digital systems and services. This dual approach to augmentation enables the DT to evolve and adapt dynamically to changing requirements and environments, effectively bridging the gap between the physical and digital worlds.

Furthermore, the augmentation can be achieved adding internal functionalities within the DT architecture allowing their interaction with the current computed State of the DT, its historical data and storage layer and on the other hand with the model to communicate the result of the augmentation function and enable its validation before the computation of a new State. The integration of these augmentations into the DT Model allows the DT to operate more autonomously, adaptively, and intelligently in response to its environment and stakeholders' needs. Moreover, the DT's relationships with both the physical and digital communication channels play a crucial role in facilitating the execution of augmented functionalities. For instance, when an augmented DT detects an anomaly in the physical world, it can communicate this information to external systems or devices through the digital communication channel, triggering appropriate actions or alerts.

From a software architecture standpoint, a DT solution should facilitate augmentation capabilities, encompassing both physical and communication aspects. This entails enabling the seamless addition of new communication protocols, interaction patterns, and data formats within the existing architecture, promoting dynamic and flexible integration. Moreover, the DT software architecture should support the declaration, execution, and monitoring of augmentation functions within the DT. It should also facilitate the interaction and integration of these functionalities with the DT's model, computed state, and data modules, ensuring access to historical DT data when necessary for augmentation capabilities.

The augmentation process should closely tie to the DT State, as the execution of augmented functionalities often involves updating the DT's internal state representation passing through the validation of the DT model.

EXAMPLE 5: When a predictive maintenance algorithm detects a potential equipment failure, it may update the DT's state to reflect the impending maintenance task, along with relevant metadata such as the predicted failure date and recommended actions.

In summary, DT augmentation represents a powerful mechanism for enhancing the capabilities and intelligence of digital twins, enabling them to operate more effectively and autonomously in complex, dynamic environments. By leveraging the DT Model, relationships with communication channels, and integration with the DT State, augmented DTs can deliver value-added services, optimize resource utilization, and improve decision-making processes across various domains and applications.

4.7 Digital Twin - Digital Representation

Having an interoperable and standard Digital Twin Representation (DTR) on the digital space shall be a fundamental principle in the design of a DT Software Architecture and results crucial for ensuring seamless integration and effective communication between DTs and external digital components such as applications, services and/or other twins. As schematically depicted in Figure 10, the DTR should be structured considering both the Digital Twin Description (DTD) and the representation its current DTS. They should be separated in two different entities or integrated into a single one according to the implementation. The first should serve as a comprehensive blueprint that encapsulates the essential attributes and characteristics of a DT, including its properties, relationships, events, and behaviours/actions. By adhering to established standards such as JSON-LD, SAREF, or other ontology-based representations, the DTD provides a structured and standardized format for describing the DT's state and capabilities. The latter should be instead responsible to represent on the digital space the current state of the DT with the values characterizing the fields described in the DTD such as properties and relationships. Actions are not mapped into current values since they are callable from the external world and events are asynchronous and should be observed.

For instance, consider a smart building management system that utilizes DTs to monitor and control various building components such as HVAC systems, lighting, and security. Each DT within the system would have a corresponding DTR describing its properties (e.g. temperature, humidity), relationships (e.g. connections to other building systems), events (e.g. equipment malfunctions), and behaviours/actions (e.g. adjusting thermostat settings) belonging to the DTD together with their values associated to the State. This DTR enables external applications, such as energy management software or occupancy tracking systems, to understand the state and functionality of each DT within the building ecosystem.

Moreover, the DTR serves as a critical link between the DT State and the computational model responsible for processing and interpreting it. The model utilizes the information provided by the DTR to compute the current state of the DT, analyse incoming data streams, and make informed decisions or trigger appropriate actions.

EXAMPLE 1: In the context of predictive maintenance, the model may utilize data from the DT State (e.g. equipment health metrics) to detect early signs of potential failures and generate maintenance alerts or recommendations.

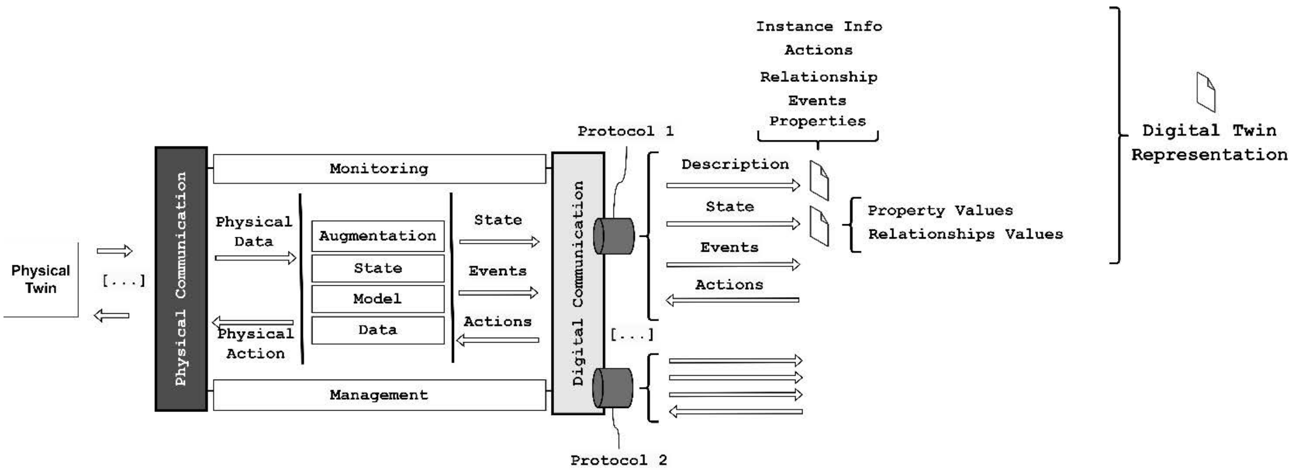

Figure 10: DT representation on the digital space through its description and the digitalization of its current state

Embedding relationships between DTs within the DTR is crucial for creating a cohesive ecosystem where DTs can collaborate and interact effectively. The DTR not only captures the attributes and characteristics of individual DTs but also defines the relationships between them, allowing for a comprehensive representation of the digital ecosystem. These relationships can encompass various types of interactions, including hierarchical, spatial, functional, or temporal connections between DTs.

EXAMPLE 2: Consider a smart city infrastructure where each building, streetlight, traffic signal, and transportation system is represented by a DT. The DTR for the smart city ecosystem would define relationships between these entities, such as containment (e.g. buildings contain rooms), spatial proximity (e.g. streetlights located on specific streets), functional dependencies (e.g. traffic signals controlled by traffic flow sensors), and temporal correlations (e.g. transportation schedules synchronized with traffic conditions).

By embedding these relationships within the DTR, digital applications can navigate the hierarchy of DTs within the ecosystem, enabling functionalities such as contextual awareness, intelligent decision-making, and automated coordination. For instance, a city management platform could leverage the relationships defined in the DTR to optimize energy usage by coordinating streetlight schedules with traffic flow patterns or to enhance public safety by synchronizing surveillance camera feeds with emergency response systems.

Overall, incorporating relationships between DTs into the DTR enhances the interoperability, scalability, and functionality of digital ecosystems, enabling seamless collaboration and synergy among interconnected DTs and digital applications. This approach facilitates the development of sophisticated and context-aware digital solutions that leverage the collective intelligence and capabilities of the entire ecosystem of Digital Twins.

In summary, the interoperable and standardized nature of the DTR facilitates seamless communication and interoperability across diverse digital platforms, enabling DTs to interact effectively within complex digital ecosystems. This integration enhances the overall functionality and utility of DTs by enabling them to leverage external data sources, interact with third-party applications, and participate in broader digital workflows and processes.

4.8 Life Cycle

In the realm of advanced technologies and intelligent systems, the lifecycle of a DT emerges as a crucial foundational element that should model and represent the different evolution of the DT through its internal phases across its evolution. The possibility to structure and model the life cycle of a DT shall be a fundamental element characterizing the architecture of a DT with the focus of addressing fundamental needs and exhibiting distinct characteristics essential for its effective deployment allowing external applications, digital services or other twins to be aware of the state of the target DT without any prior domain specific knowledge. This clause delves into the description of the main characteristics and phases of associated to the definition of a DT lifecycle with its fundamental aspects in order to provide guideline for architecture design about how the life cycle should be implemented in the different platforms with a shared and interoperable approach.

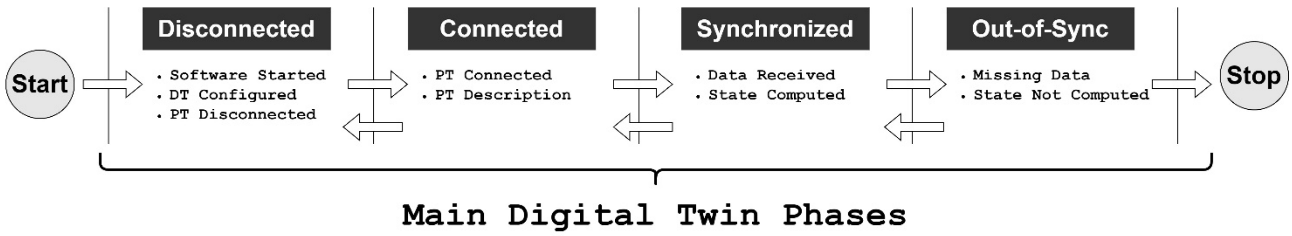

Figure 11: High level representation of the phases characterizing the DT's life cycle

The DT, serving as a dynamic bridge between the physical and digital realms, shall have a systematic and orchestrated lifecycle for several compelling reasons:

- A structured lifecycle ensures the continuous and optimal functioning of the DT, from initialization to eventual retirement, maintaining its relevance and reliability throughout its operational tenure.
- **Adaptability to Change:** The lifecycle accommodates the evolving nature of the physical asset it represents. As the real-world counterpart undergoes modifications, the DT can seamlessly adapt, ensuring a synchronized and accurate digital representation.
- **Error Resilience:** In the complex interplay between the physical and digital domains, errors and discrepancies may arise. A well-defined lifecycle incorporates mechanisms to detect, rectify, and recover from errors, ensuring the DT's robustness.

- **Resource Optimization:** Effective lifecycle management enables efficient resource allocation, ensuring that the DT utilizes computational and operational resources judiciously and remains aligned with operational goals.

Figure 11, schematically depicts the main phases that should characterize the DT lifecycle each playing a crucial role in its operation and evolution:

- **Start:** At the onset of its lifecycle, the DT is initialized, and the necessary software components are launched. This marks the beginning of its journey towards synchronization with its physical counterpart.

- **Disconnected:** Following initialization, the DT enters the disconnected phase. While the software is up and configured, the DT remains unlinked to its Physical Twin (PT), awaiting connection.

- **Connected:** Upon successful connection to the PT, the DT receives detailed descriptions of the PT's characteristics and functionalities. This exchange of information lays the groundwork for subsequent synchronization.

- **Synchronized:** As data flows from the PT to the DT, the latter accumulates enough information to compute its state. Continuously receiving and processing data, the DT maintains synchronization over time, adhering to its defined model.

- **Out-of-Sync:** Challenges may arise during the lifecycle, leading to the out-of-sync phase. This could stem from missing data or errors in computation, rendering it impossible to derive a valid state. Alternatively, received data may be insufficient or of inadequate quality for state computation.

- **Stop:** Ultimately, the lifecycle culminates in the stop phase, where the DT ceases operation. This decision may arise due to obsolescence, irrelevance, or a shift in contextual requirements, signalling the end of its utility in the given context.

The relationships and transitions among these phases and high-level phases can change across the different implementation and the aim of this clause is to identify the main steps that should characterize a DT during its implementation and evolution over time.

The DT should undergo an initialization phase, launching as a software or process, with internal components and modules booting up. Once all internal elements are activated, the twin is fully initialized and ready to start internal procedures, considering both physical and digital communication channels, alongside its internal model, storage, management, and monitoring functionalities.

Subsequently, the DT should establish a connection with its physical counterpart, facilitating bidirectional event flow and forming the basis for synchronized interactions. This phase involves the twin discovering the physical capabilities, allowing it to comprehend what is available on the PT and determining which features align with its model for digitalization and synchronization.

Upon successful association and communication initiation with associated physical counterparts (potentially multiple PTs), they should become prepared to receive data and commence the digitalization and synchronization between the physical and digital realms. Following this phase, the reflection and digitalization process should begin, with the responsibility of synchronizing the DT's state with the physical asset, enabling a dynamic (potentially real-time) reflection over time of the changes in the physical world.

During the synchronization process, potential errors such as network issues, delays, or missing information may arise. The Digital Twin's life cycle shall account for these conditions, and mechanisms should be in place for external applications or observers to receive notifications during transitions, particularly when the DT faces challenges in accurately digitalizing its physical counterpart. Resolving synchronization issues marks a phase of life cycle recovery, affirming the validity of the digitalized replica.

It is essential to note that communication with external digital entities should be facilitated through the digital communication channels of the DT. To enable this functionality, the internal management of the DT's life cycle shall be interconnected and exposed to the digital side of the DT, ensuring its availability in the digital realm. This interconnectedness ensures a seamless bridge between the physical and digital aspects of the Digital Twin, enhancing its overall effectiveness and adaptability.

At a certain point, the DT can be stopped, interrupting the synchronization state while keeping its software entity active and accessible. The stopped state marks the conclusion of its life cycle, indicating the cessation of active operations.

Throughout its life cycle, the DT maintains detailed memory and event logs, enhancing traceability and offering insights into historical events, states, and interactions.

An intrinsic characteristic of the DT life cycle is dynamic adaptability. As the physical asset evolves, the DT can adjust its state, properties, and interactions, ensuring a resilient and relevant digital representation.

In conclusion, the DT's lifecycle emerges as a critical facet that should be taken into account in the design and characterization of DT architectures and framework and a strategic approach to support twins' operational deployment, monitoring and management. The defined phases and characteristics ensure not only the seamless functioning of the DT but also its adaptability, error resilience, and optimization of resources, collectively contributing to its effectiveness in diverse operational landscapes.

4.9 Digital Twin Management

Effective management of DT is essential for ensuring their optimal performance, adaptability, and scalability within complex digital ecosystems where multiple twins can be executed, and their behaviour and execution should be managed and potentially orchestrated by authorized entities (e.g. applications, users and platforms) as schematically reported in Figure 12. The manageability of DTs should encompass various aspects, including configuration, monitoring, control, and adaptation, all of which are crucial for their seamless integration into diverse applications and environments.

One key aspect of DT manageability should be the ability to dynamically adapt the behaviour of the twin based on changing conditions and requirements. This adaptation is facilitated by the DT model and state, which provide a comprehensive understanding of the twin's characteristics, capabilities, and context.

EXAMPLE 1: Consider a smart manufacturing environment where a DT represents a production machine. By continuously monitoring its state and performance metrics, such as temperature, vibration, and throughput, the DT can dynamically adjust its operational parameters, scheduling, and maintenance routines to optimize productivity, prevent failures, and minimize downtime.

Another critical aspect of DT manageability is the configuration and management of communication channels responsible for handling digital and physical interactions. In a DT architecture, this interfaces should serve as the bridge between the DT and the external world, enabling seamless interaction with digital applications, physical assets, and other DTs. For instance, in an Industrial Internet of Things (IIoT) deployment, digital adapters facilitate communication between the DT and enterprise systems, cloud platforms, and analytics tools, while physical adapters enable connectivity with sensors, actuators, and industrial control systems on the shop floor.

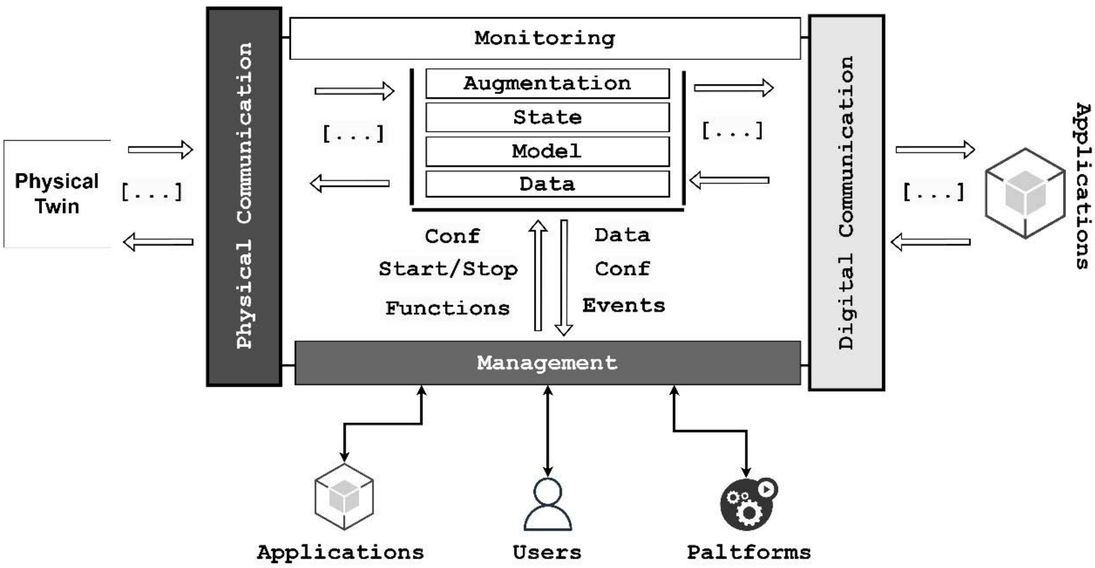

Figure 12: Interaction patterns associated with DTs management

The strategic importance of DT manageability lies in its ability to facilitate the configuration, deployment, and operation of DTs at scale while ensuring interoperability, reliability, and security. A well-managed DT ecosystem allows organizations to efficiently orchestrate their digital assets, optimize resource utilization, and respond promptly to changing business needs and market conditions. Moreover, by leveraging advanced management tools and techniques, such as edge computing platforms, automation frameworks, and machine learning algorithms, organizations can enhance the agility, resilience, and intelligence of their DT deployments.

Moreover, the DT management capabilities should encompass the ability for authorized external entities to access DT data, such as state transitions and associated events. This enables stakeholders to comprehend the underlying causes of potential anomalies. For instance, imagine a smart factory DT where authorized technicians can access data regarding equipment malfunctions and the corresponding events to diagnose and address issues promptly.

Additionally, external entities should be able to retrieve the current active configuration of the twin, encompassing both digital and physical interfaces, along with insights into the internal model and embedded augmentation functions.

EXAMPLE 2: In a smart home DT, authorized users could access the current settings of connected devices and understand how the embedded functions operate to automate household tasks.

Advanced versions of these management capabilities, if supported by the implementation and model, could facilitate dynamic configuration of new DT functionalities. This could include adjusting model behaviours or implementing planned augmentation functions in real-time. For instance, in an autonomous vehicle DT, manufacturers could remotely update the behaviour model to improve navigation algorithms or deploy new augmentation functions to enhance vehicle safety features.

EXAMPLE 3: A smart grid environment, DTs representing power generation assets, distribution networks, and consumer devices can be dynamically configured and managed to optimize energy production, distribution, and consumption. By analysing real-time data streams from smart meters, weather forecasts, and grid sensors, DTs can adjust energy generation schedules, load balancing algorithms, and demand-response strategies to minimize costs, reduce emissions, and enhance grid reliability.

In the conceptual framework depicted and structured in Figure 12, the management module is depicted as distinct and separate from the digital communication channel and functionalities of the DT. This architectural decision is deliberate, aimed at decoupling responsibilities between these two vital components. The rationale behind this separation lies in enhancing modularity and flexibility within the DT software architecture. However, while the existence of a management channel should be a fundamental element for a DT software framework, its implementation can vary. For instance, it could leverage existing capabilities of the digital communication channels, such as utilizing messaging protocols for management commands and responses. Alternatively, a standalone and independent interface could be developed specifically for management purposes, offering dedicated functionalities tailored to administrative tasks and system monitoring. This flexible approach ensures adaptability to diverse deployment scenarios and enables efficient management of DT instances across various domains and applications.

Considering another example within a healthcare use case, DTs can be deployed to monitor patient health, track medical assets, and optimize clinical workflows. By integrating with electronic health record systems, wearable devices, and medical imaging equipment, DTs can provide real-time insights into patient conditions, treatment efficacy, and resource utilization. Healthcare providers can use these insights to personalize care plans, improve patient outcomes, and streamline operational processes.

On the other hand, in a smart city environment, DTs can be used to manage and optimize transportation systems, including public transit, traffic flow, and mobility services. By aggregating data from traffic cameras, GPS sensors, and connected vehicles, DTs can predict congestion patterns, recommend alternate routes, and coordinate traffic signals to reduce congestion and improve travel times. Additionally, DTs can enable seamless integration between different modes of transportation, such as buses, trains, bikes, and ride-sharing services, to enhance urban mobility and accessibility.

In summary, the manageability of DTs plays a crucial role in maximizing their value and impact across various domains, from manufacturing and healthcare to energy and transportation. By leveraging dynamic adaptation, configuration management, and strategic planning, organizations can harness the full potential of DTs to drive innovation, efficiency, and sustainability of cyber-physical applications and deployments.

4.10 Digital Twin Monitoring

Monitoring DT instances should be essential for ensuring optimal performance, reliability, and effectiveness in achieving their intended goal and be aligned with external entities managing and monitoring twins. By collecting and analysing performance metrics from various domains, including networking, software, and operational parameters, organizations can gain valuable insights into the behaviour and functionality of DTs.

As represented in Figure 13, monitoring DTs should allow authorized entities to track Key Performance Indicators (KPIs) and metrics related both performance of the twin such as latency, throughput, resource utilization, and error rates but also to expose specific DT's aspects associated for example to the current life cycle state and/or the current entanglement level of the twin [i.4]. Furthermore, the monitoring capabilities should allow also to read and proactively observer twin's life cycle phases and logs to understand the evolution of actions, detect warning and understand errors during the execution. The aim of DTs monitoring should be to observer domain independent metrics and parameters in order to allow general-purpose observers to understand twins' performance and manager their deployment (e.g. through the management module). Any domain specific metrics (e.g. associated to IIoT production results) should be delegated to the DT's State computation and its presentation through the digital communication channel.

By analysing these parameters, organizations can identify bottlenecks, inefficiencies, and potential issues affecting DT performance and take proactive measures to address them.

EXAMPLE 1: In an IIoT deployment, monitoring sensor data ingestion rates and processing times can help optimize data pipelines and ensure timely decision-making in industrial processes. Monitoring DTs should enable DT managers (e.g. authorized users or platforms) to assess their performance against predefined goals, objectives, and Service Level Agreements (SLAs). By correlating performance metrics with business outcomes, organizations can evaluate the effectiveness of DTs in achieving their intended purposes and make data-driven decisions to improve performance.

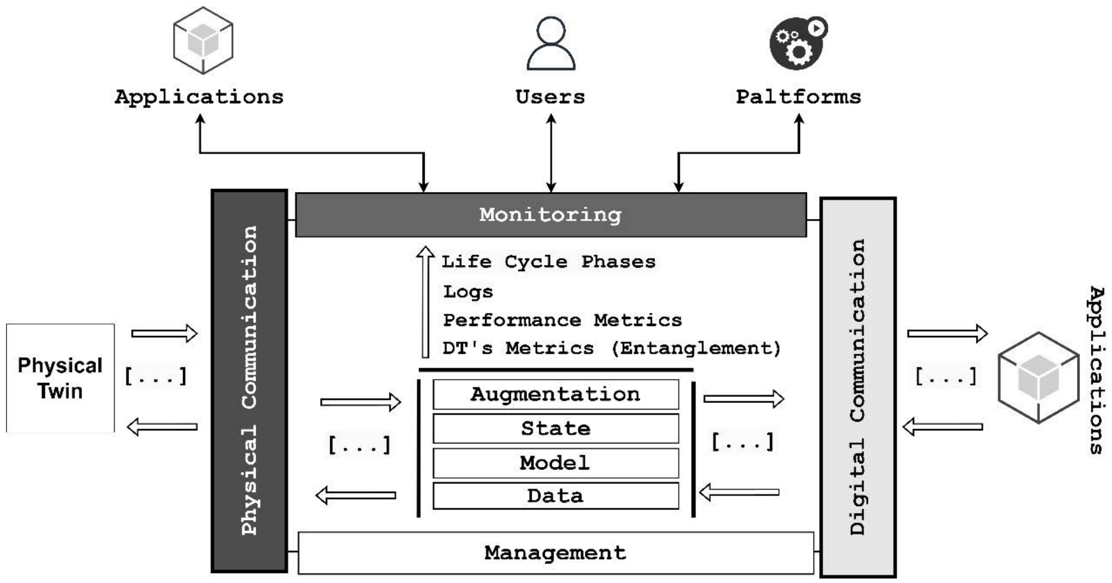

Figure 13: Interaction patterns associated to DTs monitoring and its cyber-physical characteristics

Monitoring DTs throughout their life cycle enables organizations to continuously improve their performance, reliability, and scalability. By capturing performance data over time, organizations can identify trends, patterns, and areas for optimization, leading to iterative refinements and enhancements in DT design and operation.

EXAMPLE 2: By analysing historical data on software response times and resource utilization, organizations can fine-tune algorithms, algorithms, and configurations to improve overall system performance.

Physical and digital communication channels should be structured to help the collection of performance metrics from DTs and their associated physical and digital entities. Physical communications with sensors, actuators, and industrial control systems to collect real-time data on environmental conditions, equipment status, and process parameters. Digital communication capabilities should on the other hand facilitate the communication with external systems, databases, and cloud platforms to capture operational metrics, user interactions, and application performance data.

Physical and digital layers should also support the usage of performance metrics for decision-making, analysis, and reporting purposes. By integrating with monitoring and analytics platforms, adapters enable organizations to visualize, analyse, and interpret performance data in meaningful ways, such as generating dashboards, reports, and alerts.

EXAMPLE 3: A digital adapter may integrate with a cloud-based monitoring service to track network latency, response times, and resource utilization in real-time and trigger automated actions or notifications based on predefined thresholds.

In the conceptual framework delineated in Figure 13, the monitoring module stands apart from the digital communication channel and functionalities of the DT. This strategic design choice is intentional, aiming to untangle responsibilities between these essential components. The motivation behind this segregation is to foster modularity and flexibility within the DT software architecture. While a monitoring channel should be indispensable in a DT software framework, its implementation can vary. It might capitalize on the existing capabilities of digital communication channels, employing messaging protocols for monitoring functions. Alternatively, a dedicated standalone interface could be developed explicitly for monitoring purposes, furnishing specialized functionalities tailored to administrative tasks and system surveillance. This adaptable approach ensures versatility across diverse deployment scenarios, facilitating effective management of DT instances across a myriad of domains and applications.

In conclusion, monitoring DTs for performance optimization is essential for ensuring their effectiveness and reliability in achieving their intended goals. By collecting and analysing performance metrics from various domains and leveraging physical and digital adapters for data collection and usage, organizations can continuously improve the performance, reliability, and scalability of DTs across diverse applications and environments.

5 Digital Twin Blueprint Architecture

5.0 Foreword

In this clause, the blueprint architecture of a DT is explored, considering the requirements identified in previous documents and the technical aspects highlighted in the previous clause and schematically illustrated in Figure 14. For instance, imagine a scenario where a manufacturing company wants to implement a Digital Twin system to monitor the performance of its production line. They would need to consider various factors such as the physical components of the production line, the data generated by sensors, and the digital interfaces required to interact with other systems in the factory.

The core interfaces defining the DT architecture are analysed, with a focus on the Physical Interface (PI) responsible for interacting with the physical world. Through modular adapters, the PI manages interactions with PTs and decouples complexity from the DT core.

EXAMPLE 1: In the manufacturing scenario, the Physical Interface would include adapters for connecting to sensors measuring temperature, pressure, and speed on the production line.

Conversely, the Digital Interface (DI) interacts with the digital space using modular characteristics and multiple digital adapters. These adapters facilitate communication with external applications by exposing descriptions, states, events, and triggers for DT actions through various protocols. Continuing with the manufacturing example, the Digital Interface would allow the Digital Twin system to exchange data with the factory's enterprise resource planning system or a cloud-based analytics platform.

Additionally, two interfaces have been identified: the Management Interface (MNI) and the Monitoring Interface (MOI). The MNI enables dynamic control, configuration, and management of DT behavior, acting as a digital control center. In our manufacturing scenario, the Management Interface would provide tools for administrators to configure the Digital Twin system's behavior, set performance thresholds, and allocate resources effectively.

The MOI provides insights into DT operations over time, supporting operational analytics and predictive maintenance strategies. For instance, the Monitoring Interface in the manufacturing context would collect data on equipment performance, analyse trends, and predict potential issues such as machine failures before they occur.

At the core of the DT architecture lies the DT model, responsible for shaping behavior and interacting with the DT state. Internal components utilize local data storage to store information related to events and computed states, supporting the execution of augmentation functions.

EXAMPLE 2: The DT model in the manufacturing scenario would incorporate algorithms to predict equipment failure based on historical data and adjust production schedules accordingly.

The architecture includes a dedicated augmentation function model management component and a Lifecycle Manager responsible for monitoring and computing the DT's lifecycle phase. This ensures that the Digital Twin system evolves and adapts to changes in the physical environment over time, maintaining its effectiveness and relevance.

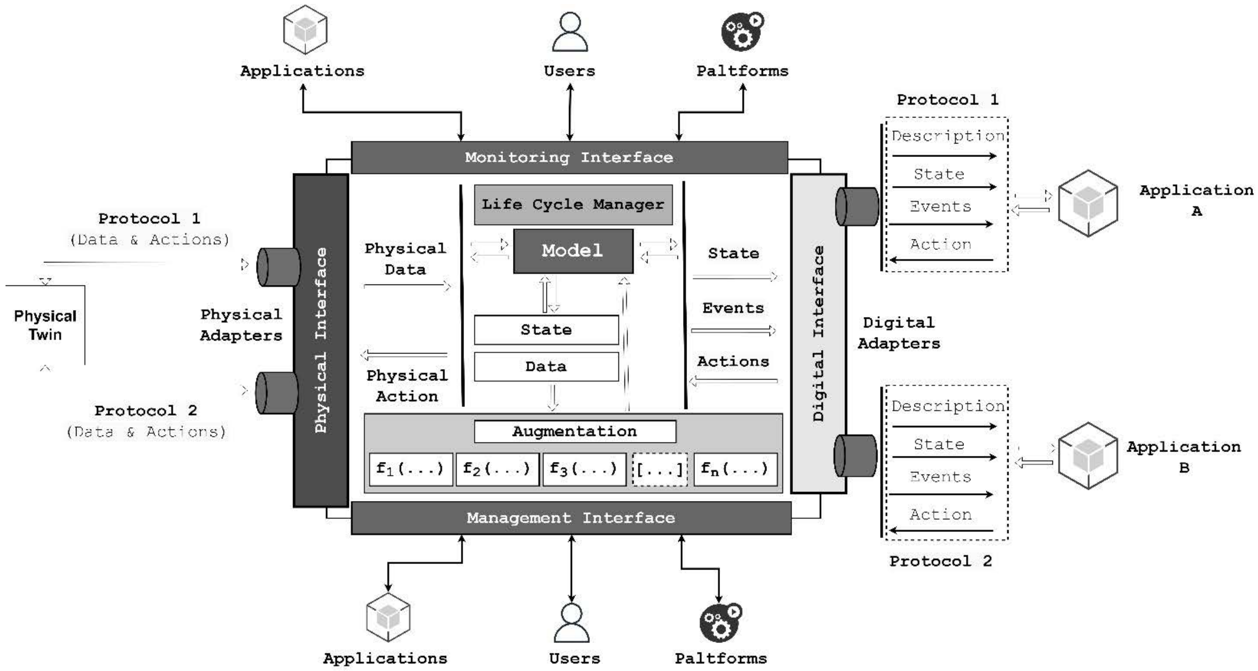

Figure 14: High level abstraction of a Digital Twin Blueprint Architecture with its main communication interfaces with physical and digital world, monitoring and management interfaces for the support of twins deployment and the internal core module for the implementation of its behavior

5.1 Physical Interface & Physical Adapters

The PI shall stand as a crucial bridge connecting the digital and physical domains within the DT architecture. It shall facilitate seamless communication, interoperability, and adaptability, enabling effective interactions between various physical twins (such as objects, devices, or assets) and their digital counterpart.

EXAMPLE: Consider a smart manufacturing environment where the PI should enable communication between machinery on the factory floor and the DT system, translating sensor data into a format understandable by the digital model. The PI should support multiple interaction forms (e.g. Pub/Sub, RESTful) and protocols (e.g. HTTP, MQTT), accommodating diverse physical entities into the DT ecosystem.

The PI should be designed through and extensible approach, equipped with versatile Physical Adapters (PA), empowers DTs to engage in meaningful exchanges with their physical counterparts. These adapters should be in charge of detecting physical events, transmit relevant information, and synchronize states, providing real-time insights and responses. For instance, in an industrial IoT setting, PA should translate signals from various sensors monitoring temperature, pressure, and humidity into a standardized format for the DT to analyse and act upon.

Furthermore, utilizing multiple adapters should allow to disentangle interface management from direct communication with the physical world, promoting reusability across different DT implementations and deployments. This flexibility extends to diverse application scenarios, such as managing manufacturing machines with different protocols or integrating sensor-rich devices with varying data formats. By harnessing adaptable adapters tailored to specific communication requirements, the PI should ensure DTs can seamlessly interact with multiple physical entities simultaneously, enhancing their reach and versatility.

In essence, the PI shall play the pivotal role in converting real-world events into digital information, aligning with the core purpose of DTs to provide actionable insights for decision-making. It enables DTs to mimic, simulate, and coexist with the physical environment by facilitating communication, synchronization, and adaptability. Ultimately, this communication interface shall serve as a vital link between the tangible and digital worlds, enriching the DT ecosystem with valuable interactions and insights, driving informed decision-making and operational efficiency.

5.2 Digital Twin Model & State

The Digital Twin Model (DTM) shall serve as the foundation for capturing and representing the PT in the digital realm with an appropriate level of abstraction associate to the DT context and the target application and deployment goals. This abstraction should help in focusing on domain-level information rather than technological intricacies, ensuring relevance and effectiveness in modelling. The link between the physical and digital replicas, known as digitalization, enables the continuous and potentially real-time updating of the digital twin's internal state in response to changes occurring in the physical asset as schematically represented in Figure 15.

Every DT should be characterized by an internal DTM shaping the representation of the corresponding PT starting from the description of the physical characteristics and functionalities coming from the PI after a discover or validation phase of the physical twin. The DTM should acts as the blueprint for the twin's behaviours, accommodating a spectrum of model complexities. At its simplest, the DTM should entail basic data validation procedures executed in response to inputs from the PT. Conversely, more intricate models should incorporate advanced techniques such as Machine Learning (ML) algorithms or sophisticated intelligent reasoning and decision-making processes.

EXAMPLE 1: Consider a smart energy grid where each building is represented by a DT. The DTM of these DTs might range from simple models that validate energy consumption data received from physical sensors to complex models that analyse historical consumption patterns, weather forecasts, and occupancy schedules to optimize energy usage in real-time.

The DTM should not only define the behaviour of the twin but also ensures the continuous updating and maintenance of the Digital Twin State (DTS), which serves as the internal representation of the twin, associated with the connected and linked PTs, throughout their life cycle and evolution. This DTS forms the foundational data structure leveraged by various DT functionalities, such as augmentation and digital interfacing, to extend capabilities and expose the DT representation in the digital domain.

For instance, in a smart manufacturing scenario, the DTM may oversee the optimization of production processes by analysing real-time data from sensors embedded in machinery. As the DTM evolves with the introduction of new insights and adaptations, the DTS reflects these changes, enabling the DT to adapt and interact seamlessly within the digital ecosystem. Additionally, the DTS should serve as the basis for creating a Digital Twin Description (DTD), which maps the DTS to the digital realm, facilitating interoperability and communication with external digital systems and applications. Through the DTD, the DT can effectively convey its current state and capabilities to the digital world, enabling informed decision-making and streamlined operations.

The DTS, maps the same information that then will be supported and mapped into the DTD and that are managed and defined through the DTM, encompassing several key components:

- **Properties:** These represent the observable attributes of the PT, such as temperature, humidity, occupancy status, energy consumption, etc. These properties are labelled data whose values can dynamically change over time, reflecting the evolving state of the physical asset. For instance, in a smart building context, properties could include temperature readings from sensors, occupancy status from motion detectors, and energy consumption data from smart meters.

- **Events:** Domain-level events that can be observed in the PT are represented within the DT State. These events capture significant occurrences or changes in the physical environment that are relevant for monitoring and analysis. In a smart building scenario, events could include fire alarms, HVAC system failures, or security breaches detected by surveillance cameras.

- **Relationships:** Links between the modelled PT and other physical assets within the organization are represented as relationships in the DT State. These relationships establish connections and dependencies between different assets, enabling holistic monitoring and management.

EXAMPLE 2: In a smart building system, relationships could define the connections between HVAC systems and individual rooms, lighting systems and occupancy sensors, or security systems and access control devices.

- **Actions:** Actions that can be invoked on the PT through interaction with the DT or directly on the DT itself, if they are not directly available on the PT, are also part of the DT State. These actions allow for remote control and management of the physical asset's functionalities, augmenting its capabilities through digital means. For instance, in a smart building environment, actions could include adjusting thermostat settings, activating/deactivating lighting fixtures, or unlocking/locking doors remotely.

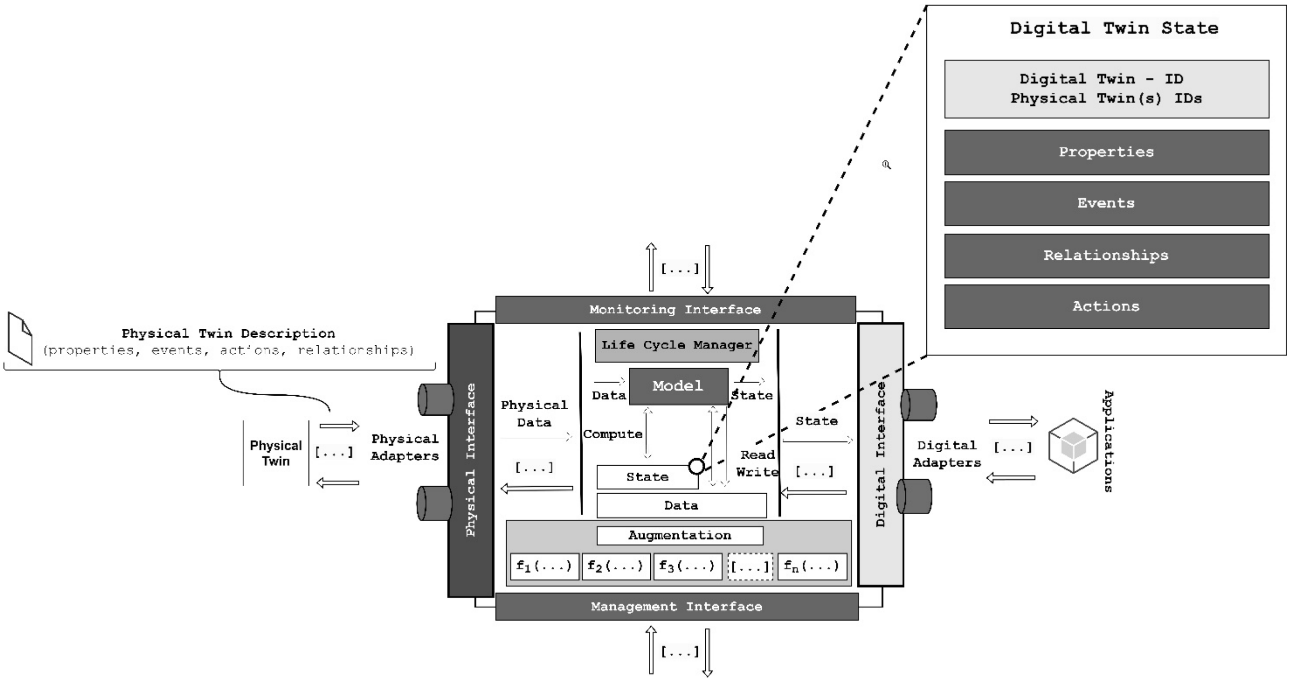

Figure 15: DT's State Management through the DTM and the relationships with physical and digital realms in this case focusing on the interaction flow from the physical to the digital

Once the model DTM is defined, the dynamic DTS should be determined by a combination of its properties, events, relationships, and actions, associated with the DT timestamp representing the current synchronization time between the physical and digital counterparts. This approach and balance between model and state enables and effective decoupling of responsibility comprehensive monitoring, analysis, and management of the physical asset within the digital ecosystem, facilitating effective decision-making and optimization of operational processes. The core process of a DT combining Model and State is the digitalization procedure detailed in the next clause with respect also to the identified blueprint architectural components and their interactions.

5.3 Digitalization Management

The process of managing DT digitalization involves the intricate coordination between the physical and digital realms to ensure synchronization and effective communication through the adoption of the Digital Twin Model (DTM) as the reference for shaping and implementing DT behaviour. As anticipated in clause 5.2, the digitalization mechanism should be in charge of maintaining the Digital Twin State (DTS) aligned with the state of its corresponding physical resource, according to the nature of the DT, its application context and the logic and behaviour defined by the DTM. At the same time, and at the same time it should also handle the incoming actions or requests from the digital world in order to validate them and forward to the PT as previously described in clause 4.5.

In the architecture facilitating the digitalization process of a DT, several components collaborate seamlessly to enable bidirectional communication between the physical and digital realms. The process entails two primary flows: the *Physical to Digital Flow*, responsible for updating the DT state based on changes in the physical asset, and the *Digital to Physical Flow*, facilitating the forwarding of digital actions to the physical asset.

In the Physical to Digital Flow (schematically illustrated in Figure 16), the journey begins with the capture of any alterations in the physical asset's state, encompassing events such as sensor readings, equipment status changes, or environmental fluctuations. These captured events should then be transmitted to the DT Model through the Physical Interface (PI) and the integrated Physical Adapters (PAs) that has the role of extracting the complexity of physical communication protocol and forward the content of the physical variation to the DTM, where they undergo interpretation and processing. Leveraging the information provided in the Physical Twin Description (PTD), the DT Model should discern the significance of these events and their implications for the digital twin's state computation. Subsequently, the DTS is updated based on this information (and/or from already stored data in the DT data module) and the logic defined within the DT Model. This update process might involve modifying existing properties, creating new ones, or triggering events within the DT. Once the new DTS has been computed, it should be forwarded to the Digital Interface (DI) where it is exposed to the external cyber world through the protocols adopted and implemented by the Digital Adapters (DAs).

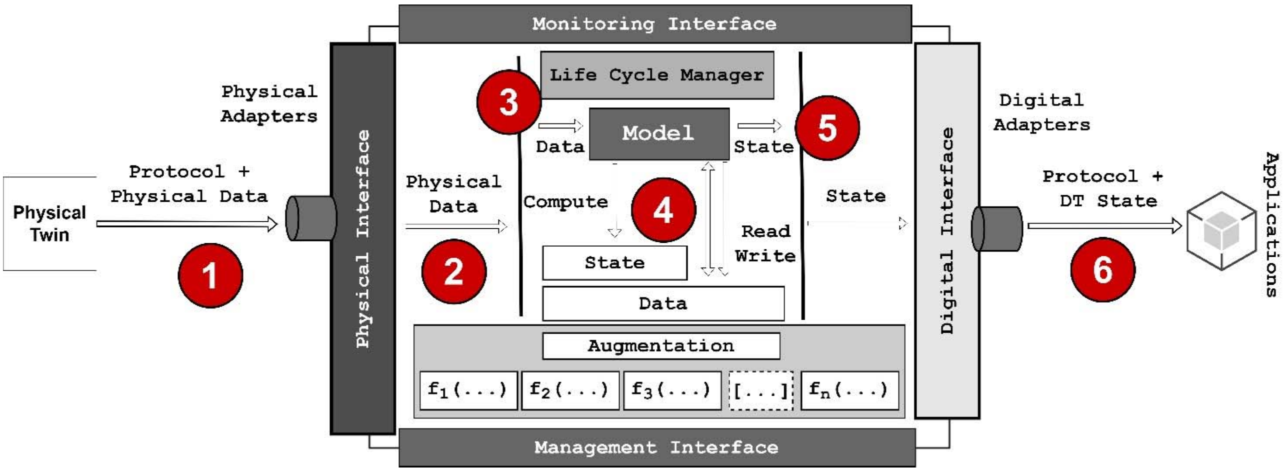

Figure 16: DT Digitalization Process from physical to digital with the associated involved architectural components

Conversely, in the Digital to Physical Flow schematically represented in Figure 17, the process should manage and unfold the receipt of a request for action from the digital world by the DT using the protocol implemented and supported by the DAs active on the DI. This request could entail commands to perform specific tasks, adjust parameters, or initiate processes. Upon receiving the request, the DTM steps in to validate its feasibility and compatibility with the physical asset and the internal behaviour (e.g. a specific action can be unavailable for a specific range of time or not compatible with the current state of DT and PT). Through scrutiny against the constraints and capabilities outlined in the PTD, the model ensures alignment between the requested action and the physical asset's capabilities. Once validated, the action is propagated from the DT to the PT through the PA on the PI associated to the correct connected PT. This step potentially involves the conversion into a format comprehensible by the physical system and transmission via the relevant communication channel. Finally, the physical twin executes the requested action, leading to tangible changes in its state, such as parameter adjustments, component activations or deactivations, or process initiations.

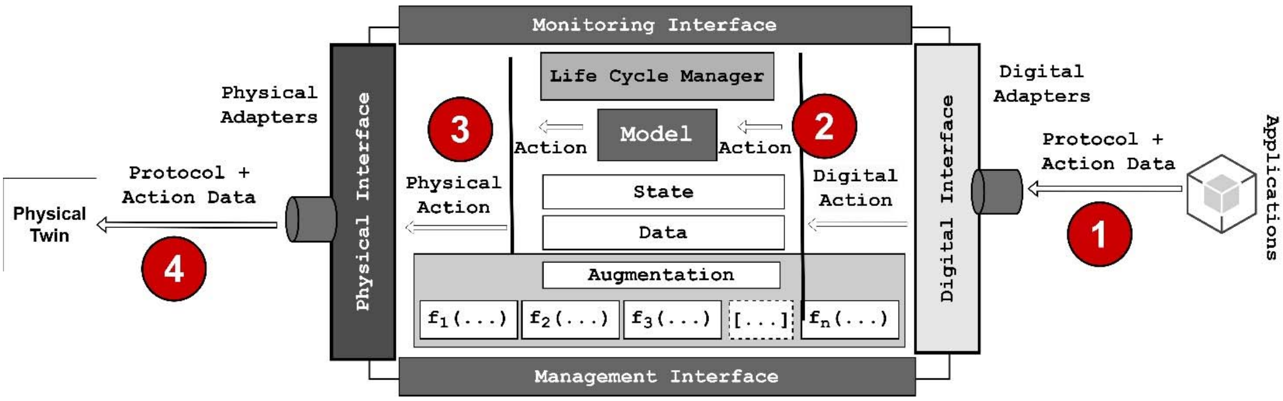

Figure 17: DT Digitalization Process from physical to digital with the associated involved architectural components

EXAMPLE: Consider a smart building scenario where a DT monitors energy consumption. When a sensor detects a sudden increase in power usage (physical event), this information is relayed to the DT, which, in turn, adjusts its state to reflect the heightened energy demand. Moreover, the digitalization process empowers the DT to not only mirror the state of the physical asset but also to execute actions on its behalf. For instance, if the DT receives a digital action request through its digital interface, it validates and forwards this request to the physical asset via its physical interface. However, it is crucial to note that the execution of digital actions does not directly alter the DT's state; rather, any changes are orchestrated through the shadowing function, maintaining the integrity of the digital representation.

In essence, this architectural framework orchestrates seamless interaction between the physical and digital domains, ensuring that the DT remains synchronized with its physical counterpart while facilitating effective communication and action execution across both realms.

5.4 Digital Interface & Digital Adapters

On the opposite end of the spectrum from the PI a DT Software architecture shall have another critical component of design denoted as Digital Interface (DI) and in charge of handling in a structured and modular way the communications with the cyber world accordingly to the nature and behaviour of the target twin (as schematically depicted in Figure 14). This interface should serve as a pivotal conduit for communication and interaction between the internal dynamics of the DT and external digital entities such as other twins, applications and services. Much like the PI facilitates interactions with the physical world, the DI should act as a gateway for conveying the DT's internal variations, events, description and state to the broader digital landscape and receiving action requests that should be processed by its internal model and, if necessary, forwarded to the associated physical counterpart.

The possibility to have a structured and modular DI represent a fundamental feature for DT Software and should allow to decouple the direct communication with digital entities for example through different protocols and data formats from the logic of the interface itself and the internal behaviour of the DT. By leveraging a variety of reusable Digital Adapters (DAs), the Digital Interface should be able to adapt and manage digital interactions and events, ensuring effective communication of the DT's essence and functionality to external applications.

EXAMPLE: Consider a smart city DT utilizing the DI to share real-time data on traffic patterns and environmental conditions with urban planning applications for optimized city management. This bidirectional mediation not only extends the DT's influence beyond its immediate boundaries but also fosters continuous dialogue and collaboration between the DT and the wider digital ecosystem.

With its resilient architecture and adaptable design based on reusable adapters, the DI strengthens the DT's adaptability by transcending interactions with individual applications to encompass diverse digital landscapes, a DT Software architecture should allow to deploy different DAs within the same twin allowing to decouple the behaviour of the DT and its DTM from its presentation capabilities on the digital space. With this approach and the associate decoupling functionalities, it is possible to reuse the same twin structure and DTM using different PAs and Das according to the target deployment and application context to adapt the twin to multiple physical and digital communication protocols and data formats. For instance, in an industrial setting, the DI seamlessly integrates with various manufacturing execution systems to ensure synchronized production processes based on real-time insights from the DT. This adaptability enables the DT to remain relevant and responsive amidst evolving application requirements, effectively separating the complexity of adapting to external digital demands from the core DT operations.

Moreover, the DI facilitates not only data transfer but also collaborative innovation by enabling different applications to interact with the DT's insights and functions according to their unique needs and logic. For instance, in the healthcare sector, the DI enables medical diagnosis applications to access patient health data from the DT and provide personalized treatment recommendations based on real-time monitoring. This native digital interoperability unlocks a multitude of application scenarios, from data analysis and predictive modelling to real-time decision-making and process optimization.

In summary, the DI should serve as a crucial bridge connecting the inner workings of the DT with external digital entities, facilitating communication dynamics, improving interoperability, enhancing adaptability, and fostering collaborative innovation. Its versatile adapters and adaptable structure ensure that the DT's insights and capabilities remain accessible, responsive, and compatible across diverse digital applications and services, thereby amplifying its value proposition within the broader digital ecosystem.

5.5 Digital Twin Augmentation Management

Augmentation represents a foundational capability of DTs, serving to amplify the functionalities of their physical counterparts via their digital representations. A DT's Software architecture should support the integration of multiple Augmentation Functions (AFs) to be implemented and executed to integrate within the twin diverse behaviours and functionalities. It is crucial to note that the execution of AFs should be coordinated within DT's lifecycle and with respect to the digitalization process and management in charge to the DTM and responsible for the computation of the DTS. Augmentation capabilities should work in parallel with the DTS computation since they are separated flow within the DT's design.

To illustrate, consider a smart building equipped with a DT. Initially, the DT architecture computes the DTS, encapsulating information about various aspects such as occupancy, temperature, and energy consumption. Subsequently, AFs can leverage this computed DTS to enact specific actions, both in the digital and physical realms.

EXAMPLE 1: If the DTS indicates an anomaly in energy consumption, an AF can be triggered to notify the building management system digitally and simultaneously adjust the HVAC system physically to optimize energy usage.

Central to this process is the Augmentation module of a DT architecture, tasked with mapping the cyber-physical relationships between the PT and DT and handling events consistently across both realms. AFs augment the computed state by introducing additional values and functionalities in a decoupled manner, ensuring flexibility and modularity within the architecture. Each AF should operate on the current DTS as input and produces a new DTS augmented with additional fields such as properties, actions, events, or relationships. These augmented fields can then be utilized by the DI with its DAs to interact with external digital entities or by subsequent AFs for further processing.

In essence, augmentation should empower DTs to evolve dynamically, adapting to changing contexts and requirements. By leveraging AFs in a decoupled manner, DTs can enhance their capabilities incrementally, ensuring seamless integration and interoperability across diverse application scenarios and domains.

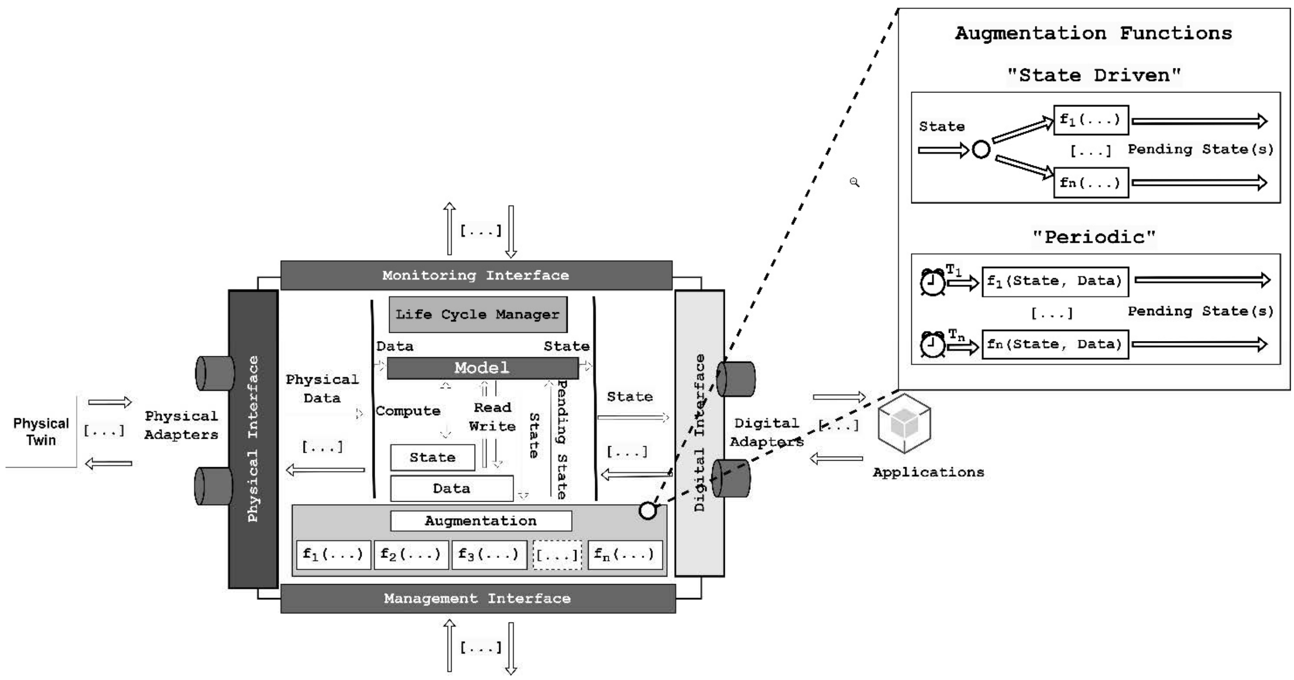

Figure 18: Schematic representation of Augmentation Functions and reference execution options and examples (state-driven and periodic)

The output of an AF should be an update of the state that should be yet validated by the DTM, and it is denoted as Pending Digital Twin State (P-DTS). The validation of P-DTSs should be always in charge of the DTM as the core element of the twin in charge of modelling its behaviour and compute its final DTS using both information from the physical world and from AFs. Following this structure and approach (illustrated in Figure 18), within a blueprint DT architecture two main categories of AFs should be mainly identified and shaped as reference categories (additional could be added within the same structure):

- ***State-Driven Augmentation Function:*** Are AFs that are executed and scheduled (also automatically according to the implementation of the DT) only when a new DTS has been computed by the digitalization process and the DTM. The AF should receive the new DTS and integrates, if necessary, the computation with data coming from the history of the twin through its internal storage module. Multiple state-driven AFs can be triggered after the computation of a new DTS and the synchronization of this functions (e.g. if they are executed in parallel or as a chain of functions) is delegated to the DTM and the specific implementation and application logic of each twin.

- ***Periodic Augmentation Function:*** In this case AFs are executed periodically by the DT's core (e.g. every minute or at a specific time schedule) and use as input the DTS as the latest computed DT state and available data from the history and storage of the DT. The output will be a new P-DTS that should be then processed and validated by the DTM for an additional consistency validation before publishing the new DTS.

In the context of DT's augmentation capabilities and functionalities the main requirements should be that DT architectures and implementation shall guarantee the consistency of the DTS across its life-cycle and after the execution of augmentation functions through the integration and communication with internal components and in particular with the DTM as the reference entity within the twin in charge of computing the final updated DTS.

EXAMPLE 2: If different AFs works on different independent DTS fields (e.g. two different properties) they might be executed in parallel without affecting the consistency of the new computed DTS. On the other hand, if two AFs might impact same shared fields and capabilities of the DT they should be synchronized and coordinated to avoid inconsistency in the digital representation and its functionalities. The validation of augmentation function results and outputs in terms of produced DTSs and their validity should be integrated and implemented in the DTM before publishing and declaring a DTS as the new valid digital representation.

5.6 Digital Twin Description

A critical aspect of designing and implementing DT architectures should the precise definition, description, and discoverability of twins within one or multiple deployments. In this context, the definition and support for a Digital Twin Description (DTD) should play a pivotal role in harnessing the full potential of DTs by providing clear guidelines and robust mechanisms for describing and discovering twins within the same domain and across diverse application domains and use cases through its potential mapping across different standards at the same time thanks to the flexibility provided by the DI with its adapters.

The formulation of DTDs should entail various prerequisites and criteria that govern their construction. These requirements should encompass multiple aspects, including identity, metadata, state descriptions, interfaces, and more as showed in Figure 19. For instance, each DT should possess a globally unique identifier, which is reported in the DTD to provide an immediate reference to the target twin. Additionally, the DTD should include identifiers of associated PTs, along with type, description, and metadata, aiding in the deployment, discovery, and understanding of the DT's capabilities and relationships. Furthermore, the DTD should integrate all the required information for an external client to access to the DT's data and information such as the DTS or invoking an exposed action according to the protocol(s) supported by the DI.

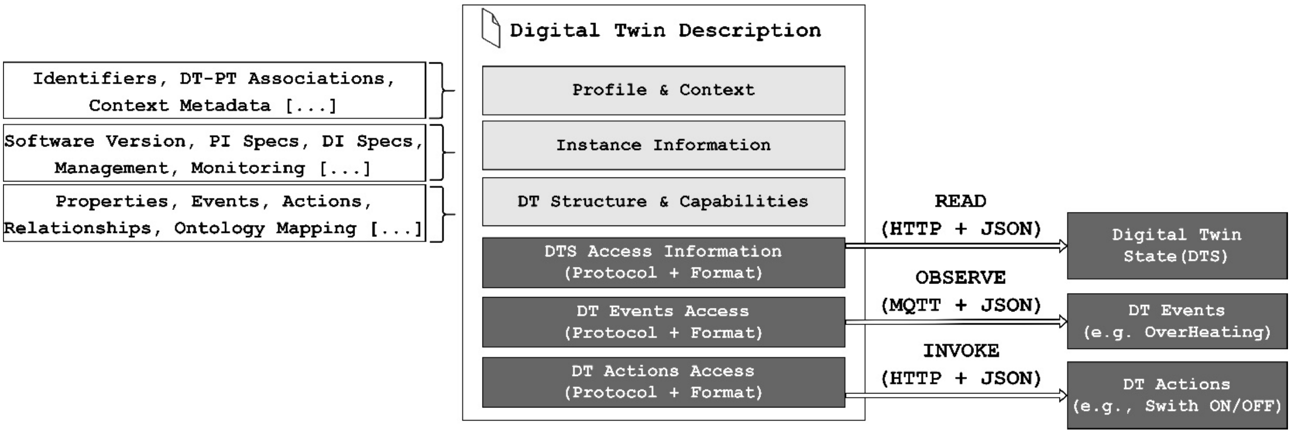

Figure 19: Illustrative representation of the DTD, its main components and the relationships with other DT-related entities

Properties, relationships, events, actions, and state descriptions and how to access that information are fundamental components of the DTD. Properties represent observable attributes of the DT, such as temperature readings or equipment status.

EXAMPLE: In a smart city DT, properties could include real-time traffic conditions, pollution levels, and energy consumption statistics.

Relationships denote connections between the DT and other entities, conveying semantic meanings like containment or association. In an industrial setting, a DT of a manufacturing robot might have relationships with DTs of other robots in the assembly line, indicating coordination and workflow dependencies.

The DTD should also encompass all ontology attributes necessary for an external application to comprehend the capabilities of the target DT. This can be achieved by integrating established ontologies, such as the SAREF Ontology for smart appliances, or other domain-specific approaches.

These ontological frameworks provide a standardized vocabulary and semantic structure that facilitate a clear understanding of the DT's functions, relationships, and properties. By leveraging the capabilities provided by the Digital Interface, the DTD can expose these ontology attributes to external applications, ensuring seamless interoperability and enhancing the ability of various systems to effectively interact with and utilize the DT. This integration not only promotes a unified understanding across different platforms but also enables more sophisticated interactions and decision-making processes based on the DT's comprehensive digital representation.

Moreover, DTDs introduce functionalities for monitoring and management of DTs. Monitoring information allows external applications to observe available metrics and events generated by the DT, enhancing accountability and awareness of the cyber-physical relationship. Management specifications enable external applications to discover and manage the DT's software instance, facilitating lifecycle control, versioning, and dynamic configurations.

The DTD should be crucial for decoupling the representation of a DT from its internal entities, such as the DTS, and managing its interactions with the external digital world. It simplifies how to access event observations, invoke actions, and read contextual information related to a digital twin instance and its environment. The DTD should comprise several key modules and components:

- **Profile & Context:** This part contains and describes the identifiers of the DT and the associated PT. It defines the cyber-physical relationship between the DT and PTs, along with additional metadata pertinent to the specific context, application domain, and use case.
- **Instance Information:** This component details the current software running the DT model and behaviour. It includes information such as the software version, the specification of PI the associated adapters, characteristics of the DI and its adapters together with management and monitoring capabilities.
- **Structure & Capabilities:** This part describes the DT's structure and capabilities, enabling external applications to understand its properties, events, actions, and relationships. These elements can be mapped into target domains using ontology attributes, such as the SAREF Ontology for smart appliances or other domain-specific frameworks, to enhance interoperability and comprehension.

- **Interaction Protocols & Data Formats:** Beyond mere descriptions, the DTD should provide information on the protocols and data formats for accessing and interacting with the DT. This includes how to read the DTS, observe generated events, and invoke actions. This ensures that external applications can effectively communicate with and utilize the DT's capabilities.

By integrating these aspects, a DTD offers a comprehensive and accessible framework that encapsulates the essential details and functionalities of a digital twin. It supports multiple descriptions and interactions based on the DT's nature and deployment use case, promoting seamless integration and interoperability within the digital ecosystem.

Effective discoverability of DTDs is crucial for leveraging DTs efficiently across diverse industries and domains. Therefore, methodologies, protocols, and techniques are explored to guide the design and implementation of DT architectures, promoting seamless exploration and retrieval of DTDs to enhance interoperability and integration.

5.7 Life Cycle Management

The design and implementation of a DT Software Architecture should include also the definition and characterization of its life cycle. As previously anticipated and introduced in clause 4.8, at least with five states and phases of a DT's life cycle should be identified with respect also to the presented blueprint architecture through which the DT goes from when it is executed to when it is stopped.

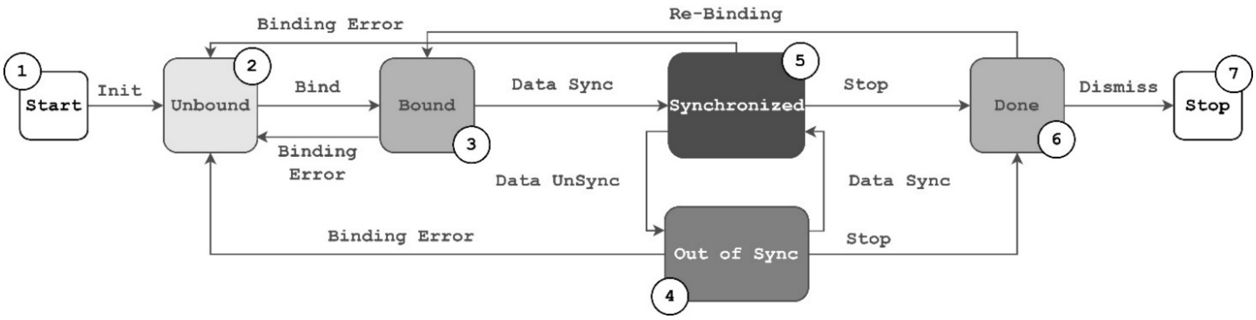

Figure 20: DT's Life Cycle phases identified in the proposed blueprint architecture

The schema presented in Figure 20 shows a graphical representation of the DT life cycle, taking into account the component of the structured blueprint architecture and encompassing the following phases:

1) **UnBound:** In this initial phase, the DT is active and all its internal modules are operational. However, it has not yet established an association with the corresponding PT. During this stage, the DT is essentially in a standby mode, awaiting the binding process that will link it to its physical counterpart. Physical In this phase, The PI is initialized but not yet active in terms of communication with the PT. The DTM is operational, defining the DT's structure and behaviour, but it has no active connection to a PT. The DTS is prepared and initialized, ready to store state information, but it remains static and not reflective of any physical entity. The DI is active and can interact with external applications, but only for internal testing or initial setup since it is not yet linked to a PT.

2) **Bound:** In this phase, the DT transitions to an active state following the successful execution of the binding procedure and starting the digitalization process by receiving data from the physical world. The binding procedure establishes a connection between the DT and the PT, enabling a bidirectional flow of events and interaction (e.g. discovering PT capabilities and reading telemetry data). The DT can now apply its model to compute its DTS by receiving updates from the PT and sending commands or updates back to it, forming a dynamic interaction loop. The PI engages in the binding procedure, establishing communication with the PT and enabling bidirectional data flow. The DTM updates to reflect the new connection, setting up rules and mappings necessary for interaction between the DT and PT. The DTS starts receiving initial data from the PT through the PI, beginning to build an accurate state representation. The DI updates to reflect the bound state, allowing external applications to begin querying or interacting with the DT based on its new connection to the PT.

3) **Synchronized:** Upon entering this phase, the DT successfully the digitalization process, ensuring that its DTS is accurately synchronized with the PT. This means the DT's digital representation reflects the status and conditions of the physical asset according to the DTM. The PI continuously transmits fresh and live data from the PT to the DT, ensuring up-to-date state information. The DTM manages the synchronization and digitalization rules, ensuring that the DTS accurately reflects the PT's current state according to the received physical information. The DTS is actively updated and synchronized with the PT, providing a fresh or even real-time digital representation. The DI facilitates external interactions, providing accurate, real-time data and state information to applications and services.

4) **Out of Sync:** This phase indicates that there are errors or disruptions in the digitalization process. When in this state, the DT is unable to maintain state alignment with the PT due to different reasons affecting both the entanglement level (e.g. connectivity issues or increased delay) or the fidelity and quality of the digital representation (e.g. wrong or corrupted data or low quality received information). Automatic or manual troubleshooting and corrective actions are required to restore synchronization, or the DT should wait until external conditions recover to a stable state. External digital entities (e.g. applications) should be aware about this state and about the fact that the DT at this step is not a correct representation of the associated PT. The PI encounters issues, such as communication errors or data discrepancies, preventing accurate data transmission. The DTM detects synchronization problems and may initiate corrective actions or alerts. The DTS becomes outdated or inaccurate due to failed or delayed updates from the PI. The DI reflects the out-of-sync state, possibly restricting certain operations or alerting external applications to the synchronization issue.

5) **Done:** This final phase occurs when the digitalization process is deliberately stopped, but the DT remains active. In this state, the DT continues to handle requests from external applications, providing data or responding to commands as needed, but it no longer updates its state based on the PT. The PI ceases active data transmission from the PT but remains capable of occasional or on-demand communication if needed. The DTM transitions to a maintenance mode, managing the DT without live updates from the PT. The DTS maintains the last known state information together with its history (if enabled and supported), but no longer new updates. The DI continues to serve external applications, providing historical or static data and supporting management requests, even though live updates are halted.

Each of these phases represents a critical step in the DT life cycle, ensuring that the DT can effectively mirror, manage, and interact with its physical counterpart across various operational scenarios.

Thoroughly examining the transitions, along with the corresponding requirements and characteristics of each phase, is fundamental for a comprehensive understanding of the DT life cycle and its associated evolution and changes. This detailed analysis is crucial for identifying the specific capabilities that should be incorporated into a DT architecture and framework, particularly those that support life cycle management. Furthermore, it enhances our understanding of cyber-physical interactions, ensuring that the DT is well-prepared to navigate and adapt to the complexities of its operational states. The phase transitions considered are the following:

From Unbound to Bound

Considering the target reference Life Cycle the first point to address is to characterize the steps that should be taken into account in order to move from an *UnBound* state to a *Bound* condition with respect to the relationship with the physical layer.

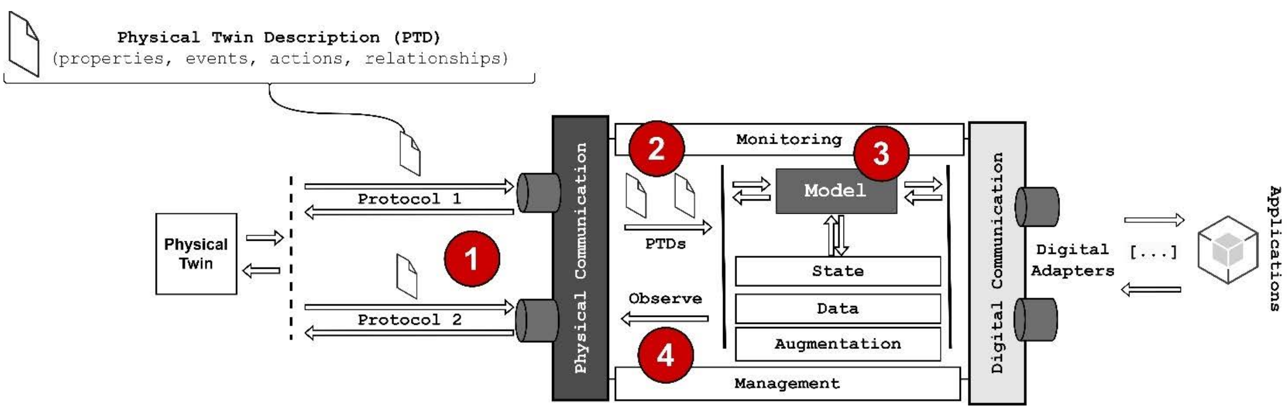

Figure 21: DT Life Cycle from UnBound to Bound

Schema in Figure 21 illustrates a simple scenario where a PT uses two protocols (P1 and P2) to communicate, and it is connected to the DT through a its PI enabled with two dedicated adapters for protocol P1 and P2. To move from the *Unbound* to *Bound* state the DT should be aware of the description of the target asset with respect to the two protocols.

EXAMPLE: Through P1 the asset exposes telemetry data (e.g. light bulb status and energy consumption) while on P2 allows incoming action requests (e.g. turn on/off the light). The DT can start the digitalization process only when it is bound and has a description of the properties and capabilities of the associated physical counterpart. Involved steps are:

1) The Adapter P1 communicates with the PT through Protocol 1 to receive or compute the associated PTD describing the capabilities exposed through the target adopted protocol. The Adapter P2 communicates with the PT through Protocol 2 to retrieve the second PTD associated to its perspective and the associated exposed data and capabilities.

2) Retrieved PTDs are sent to the DTM through the PI.

3) The DTM analyses the received PTDs and defines which characteristics (properties, events, actions and relationships) should be digitalized and mapped in its DTS and starts monitoring them through the target adapters of the PI.

4) The DTM triggers the PI to start observing the PT and receive relevant information about the physical world.

Only when all the physical adapters have been correctly bound (it may require time) to the PTs and the associated PTDs have been generated, the DT can move from *UnBound* to *Bound* having all the required information to decide which information are required by the DTM, start collecting data and trigger the digitalization process.

From Bound to Synchronized

Following the same approach described in the previous step, this second example aims to identify the steps that should be considered to define a procedure allowing the DT to move from a *Bound* state to a *Synchronized* condition where the twin identified the interesting capabilities of the PT that has to be digitalized and according to the received PTDs start the digitalization procedure to be synchronized with the physical world and its PT.

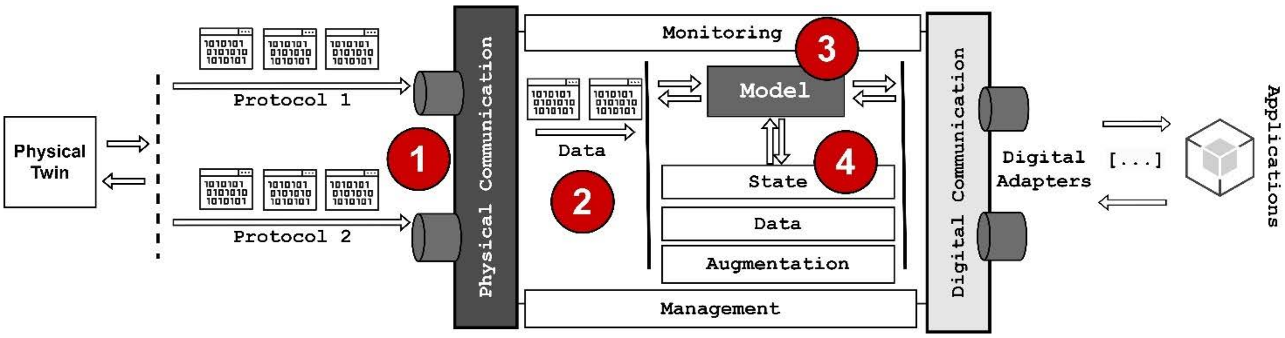

Figure 22: DT Life Cycle from Bound to Synchronized

As schematically illustrated in Figure 22, involved steps are:

1) The PI receives PT data according to the observation specification received by the DTM using both Protocol 1 and Protocol 2.

2) The PI forwards received data and the valuable payload by removing the useless protocol information.

3) The DTM receives and analyses received data within its digitalization process in order to compute and maintain updated the DTS.

4) Once the DTS has been compute it is saved as the latest fresh representation of the PT through the DT and can be used by the DI to expose the twin on the digital space.

The DT moves from the *Bound* to *Synchronized* phase until it is able to maintain a proper synchronization with the physical asset over time through its digitalization process and the generation and maintenance of the DTS.

5.8 Management Interface

At the core of DT management and orchestration a Management Interface (MNI) should be present as a fundamental element that empowers administrators and operators to dynamically control, configure, and manage the behaviour of running DT instances throughout their deployments. This interface should be in charge of enabling dynamic adaptability, efficient resource allocation, and seamless coordination and synchronization between the digital and physical realms.

The MNI should serves as a digital control centre from which administrators navigate the operational landscape of DTs. It should allow various configurable options, allowing administrators to customize the DT's behaviour in terms of both PI and DI adapters execution and their configuration, DT's behaviour, performance thresholds, and resource allocation. This dynamic governance fosters proactive decision-making, facilitating swift adjustments to DT configurations and behaviours to accommodate changing contextual requirements. This supports a vision where DTs are deployed in a structured ecosystem, managed, and monitored throughout their life cycle to align with both application goals and context variations over time.

The MNI should oversees, facilitates, and streamlines the synchronization process between the DT and its physical counterpart within the cyber-physical realm. For a DT to effectively operate, it needs to continuously receive and process real-time data from the physical environment through the PI. This data is used to construct a digital replica of the physical entity, computed through the DTM and maintained with the DTS, which is subsequently exposed to the external digital world via the DI. The MNI manages the configurations and executions of both the PI and DI, utilizing the necessary physical and digital adapters based on the specific use case and the characteristics of the connected physical and digital entities.

In an ever-changing operational landscape, the MNI should also provide fundamental mechanisms for live adaptability. Whether responding to shifts in environmental conditions, variations in performance, or unexpected challenges, the interface should enable prompt modifications to the DT's behaviour. This adaptability is vital for maintaining operational efficiency and predictive accuracy, ensuring that the DT remains a dependable representation of its physical counterpart. The MNI should oversee managing the execution of physical and digital adapters, which can be added or removed as needed to respond to significant variations in the operational context and cyber-physical requirements. This should also include the coordination with the MOI to continuously assess the DT's performance and state, ensuring that adjustments are based on accurate and current data.

The MNI should also interact with the internal storage component, ensuring that all configurations, state changes, and operational data are logged and stored for historical analysis and future reference. This interaction is crucial for maintaining a comprehensive record of the DT's lifecycle, facilitating troubleshooting, and supporting long-term planning and optimization efforts.

In essence, the MNI should serve as the central point of control, coordination, and adaptability within the digital representation of the physical world. It empowers administrators and orchestration platforms to guide DT instance behaviour, optimize resource allocation, synchronize operations, and ensure real-time responsiveness to changing contexts. As the orchestrator of the DT's dynamic evolution, this interface enhances the DT's ability to navigate a dynamic operational landscape while maintaining fidelity to the physical entities they represent.

5.9 Monitoring Interface

At the core of DT design should be taken into account also the Monitoring Interface (MOI), a structural and fundamental element that should provide insight into how DTs operate over time during their cyber-physical lifecycle. This interface should be all about capturing, conveying, and analysing a wealth of data and metrics that shed light on the DT's performance, health, and interactions with its physical counterpart.

The MOI acts as a vigilant observer, constantly collecting real-time data on the DT's performance, resource usage, and interactions. This information gives a complete picture of how the DT is performing, helping administrators assess how well it mirrors the physical asset. This empowers proactive decision-making by revealing potential areas for improvement or identifying issues.

The MOI's role extends beyond just raw data collection; it pulls out metrics that provide insights into the DT's operational context. This context-awareness boosts the accuracy of the DT's digital representation, ensuring it stays in sync with the ever-changing physical world. By analysing metrics that show how closely the DT mimics the physical asset, the MOI ensures the DT accurately reflects real-world behaviour.

The MOI communicates seamlessly with other DT architectural components to maintain this accuracy and provide comprehensive monitoring:

- **Physical Interface (PI):** The MOI collects data from the PI, which serves as the conduit for real-time information from the physical asset. This includes telemetry data, status updates, and events.
- **Model:** The MOI leverages the DT model to understand the expected behaviour and structure of the DT. This helps in contextualizing the data and identifying deviations from the norm.
- **Storage:** Collected data is stored and managed, providing a historical record for trend analysis, anomaly detection, and predictive analytics.
- **State:** The MOI continuously updates the DT state with the latest information, ensuring the digital replica is an accurate reflection of the physical asset at any given moment.
- **Digital Interface (DI):** The MOI works with the DI to expose relevant data and metrics to external applications and stakeholders, ensuring transparency and accessibility.
- **Management Interface (MNI):** The MOI collaborates with the MNI to adjust configurations and optimize performance based on the monitored data. It provides feedback for dynamic control and management decisions.

The MOI should not be limited to providing a static snapshot of DT performance; it is also a dynamic hub for operational analytics. This allows to spot patterns, trends, and anomalies over time about DT performance.

EXAMPLE: It can be used in combination with machine learning and data analysis approaches, the data coming from the MOI can be used to find potential inefficiencies, predict upcoming issues, and support data-driven decisions to optimize DT behaviour. Furthermore, advanced versions of the MOI can also internally include those functionalities allowing external applications to be notified about detected performance anomalies. The data collected by the MOI also contributes to predictive maintenance strategies. By spotting potential issues before they become major problems, it enables administrators to make informed decisions about resources, configurations, and other adaptations to keep performance at its best.

The transparency offered by the MOI enhances the accountability and trustworthiness of the DT. By providing a clear view of the DT's behaviour, actions, and interactions, the interface promotes accountability in decision-making. This transparency is crucial for building trust among stakeholders and ensuring the DT functions reliably within its operational context.

In a nutshell, the MOI should serve as a dynamic observatory that captures the essence of a DT's existence and interactions. Through real-time performance assessment, contextual insights, operational analytics, and proactive maintenance support, the interface empowers administrators to make informed decisions, fine-tune DT behaviour, and maintain trustworthiness. It acts as a bridge between the digital and physical realms, facilitating an ongoing conversation between the two and enhancing the DT's effectiveness within the broader operational landscape.

6 Digital Twins Adoption & Deployment

6.0 Foreword

DTs, as transformative entities, find varied adoption strategies and deployment architectures based on the specific needs and contexts of different domains [i.10]. This clause explores key aspects of DTs adoption and deployment, shedding light on distributed and centralized approaches, considerations for edge and cloud deployments, the integration of cyber-physical awareness, effective monitoring and management practices, and the crucial aspect of DT discoverability envisioning an ecosystem of different and interconnected DTs [i.5].

6.1 Distributed & Centralized Approaches

The adoption of DTs often involves critical decisions regarding the architectural approach. Organizations may opt for distributed or centralized models based on factors such as system complexity, scalability requirements, and data governance. Distributed approaches allow to distribute the computational load across multiple nodes, fostering resilience and scalability, while centralized approaches concentrate processing and control within a singular entity, streamlining management and governance as depicted in Figure 23. This clause explores the nuances of these approaches, providing insights into their respective advantages and considerations with respect to the introduced architectural blueprint and its main components.

When implementing DTs, architectural decisions play a crucial role in shaping the system's capabilities and scalability. Organizations have the flexibility to choose between distributed and centralized models based on their specific needs and requirements of the application domain or the target use case. Furthermore, even if different implementations will be available considering both centralized and decentralized solutions, the interoperability among them and the sharing of common principles and components should be a shared principle through different deployments and DT driven applications.

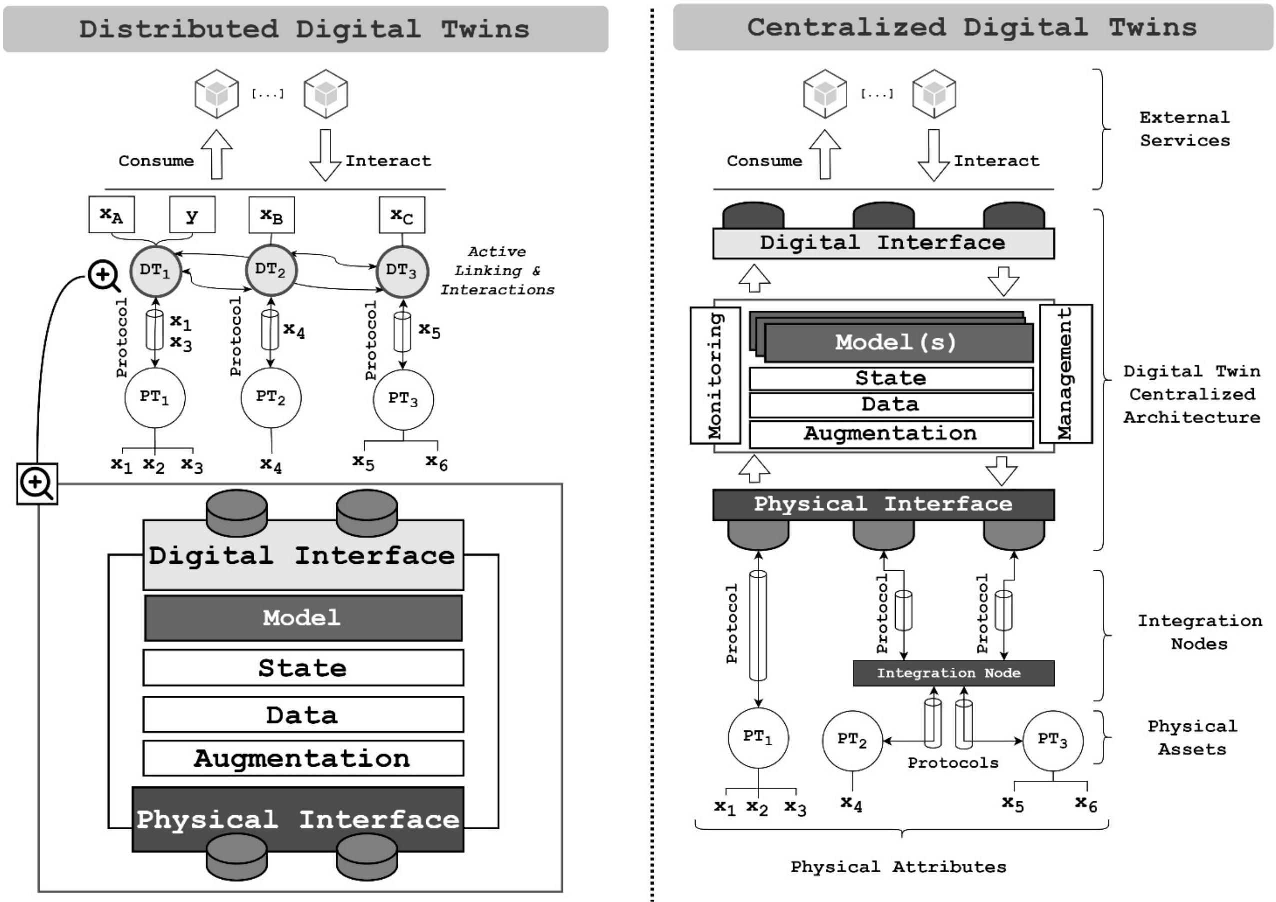

Figure 23: Schematic representation of the Decentralized and Centralized DT architecture and implementation approaches

Distributed architectures, such as microservices-based implementations, distribute computational tasks across multiple independent nodes. This approach offers advantages in terms of resilience, scalability, and fault tolerance. Each DT instance should function as an independent microservice and software component, responsible for managing its own operations and interactions and digitalizing a reference PT or a group of PTs.

EXAMPLE: In a smart city deployment, each building can have its own DT instance, allowing for localized control and monitoring. From an architectural point of view:

- **Physical Interface:** Each microservice can handle specific physical adapter for its PI implementation, ensuring dedicated and optimized interactions with PTs. If different DT instances share the same implementation adapter can be reused across different twins.

- **Model:** Distributed DTMs can be specialized for different twin according to their application goal and behaviour, enhancing the accuracy and relevance of the target twin's representation. Each DT should be in charge of digitalizing and handling its specific context and its cyber-physical relationship between itself and the associated PTs.
- **Storage:** On the one hand, each DT instance should be able to handle its internal storage capabilities while on the other hand, distributed storage solutions should allow data to be stored locally on each computing node supporting multiple twins enhancing access speed and resilience.
- **State:** Each DT should maintain its own state, which should synchronize with the associated PTs according to the internal DTM, ensuring consistency and reliability.
- **Digital Interface:** Each DT modelled as an independent microservice should expose its own DI with its set of digital adapters, enabling flexible and modular integration with external systems. As previously mentioned for the PI also in this case within a shared implementation platform, digital adapters can be reused across different twin instance.
- **Management & Monitoring Interfaces:** Each DT instance should be monitored and managed independently, providing granular insights into the performance and health of each twin instance. In order to simplify distributed monitoring and management, dedicated orchestration techniques can be adopted to coordinated groups of DTs with similar characteristics.

On the other hand, centralized architectures, such as monolithic designs, consolidate processing and control within a single entity. This approach simplifies management and governance, as all DTs instances are managed within a unified framework sharing a set of common architectural components. In a centralized setup, a monolithic application could manage multiple twins simultaneously, coordinating their operations and interactions. For instance, in a manufacturing plant, a centralized DT system could oversee and optimize the operations of various production lines and equipment. From an architectural point of view the interaction and responsibilities within centralized DT architectures should be the following:

- **Physical Interface:** Centralized DT architectures and implementation should have a unified PI, streamlining the integration process with PTs allowing the shared use of different physical adapters to enable the communication through different protocols and data formats.
- **Model:** Different DTs models and DTMs should be supported by the architecture within the same centralized solution allowing the implementation of the different twins behaviours and their mapping and digitalization of the associated PTs and ensuring consistency and standardization across all DT instances.
- **Storage:** A centralized storage system should offer a streamlined data management and easy access to a consolidated data pool allowing the structured and controlled access to stored data of the different managed twins within the architecture.
- **State:** The central entity should maintain a comprehensive and accessible representation of all the DTSs for all DTs instances managed by the solution, ensuring coherence and synchronization.
- **Digital Interface:** A single, unified DI should be available in the architecture with the aim to simplify the integration with external applications and systems using different digital adapters shared across the different DT instances running on the platform.
- **Management & Monitoring Interfaces:** Centralized management monitoring modules should provide a holistic view of all the DT instances running taking into account their performance and configurability, enabling easier management and optimization.

Despite the differences in architectural approaches, adherence to common principles, requirements, and blueprint components is essential to ensure interoperability among solutions, implementations, and deployments. Regardless of whether a DT is implemented as a distributed microservices architecture or a centralized monolithic system, it should still adhere to standardized interfaces, data formats, and communication protocols. This consistency enables seamless integration and interoperability across different DT instances, allowing them to communicate, share data, and collaborate effectively. By following a standardized blueprint architecture and principles, organizations can future-proof their DT deployments, ensuring scalability, flexibility, and interoperability across diverse use cases and environments. Whether distributed or centralized, twin can leverage common frameworks and standards to unlock their full potential in driving innovation and efficiency across various industries and domains.

6.2 Edge & Cloud Deployments

The deployment landscape for DTs spans a spectrum from edge to cloud environments as illustrated in Figure 24. Edge deployments and Edge DTs [i.8] involve placing computational resources closer to the physical assets, reducing latency and enabling real-time processing. Conversely, cloud deployments leverage the vast computational power and storage capabilities of cloud infrastructure, facilitating scalability and accessibility. This clause delves into the architectural considerations and evaluations with respect to both edge and cloud deployments, highlighting scenarios where each approach excels and the potential synergies in hybrid deployments.

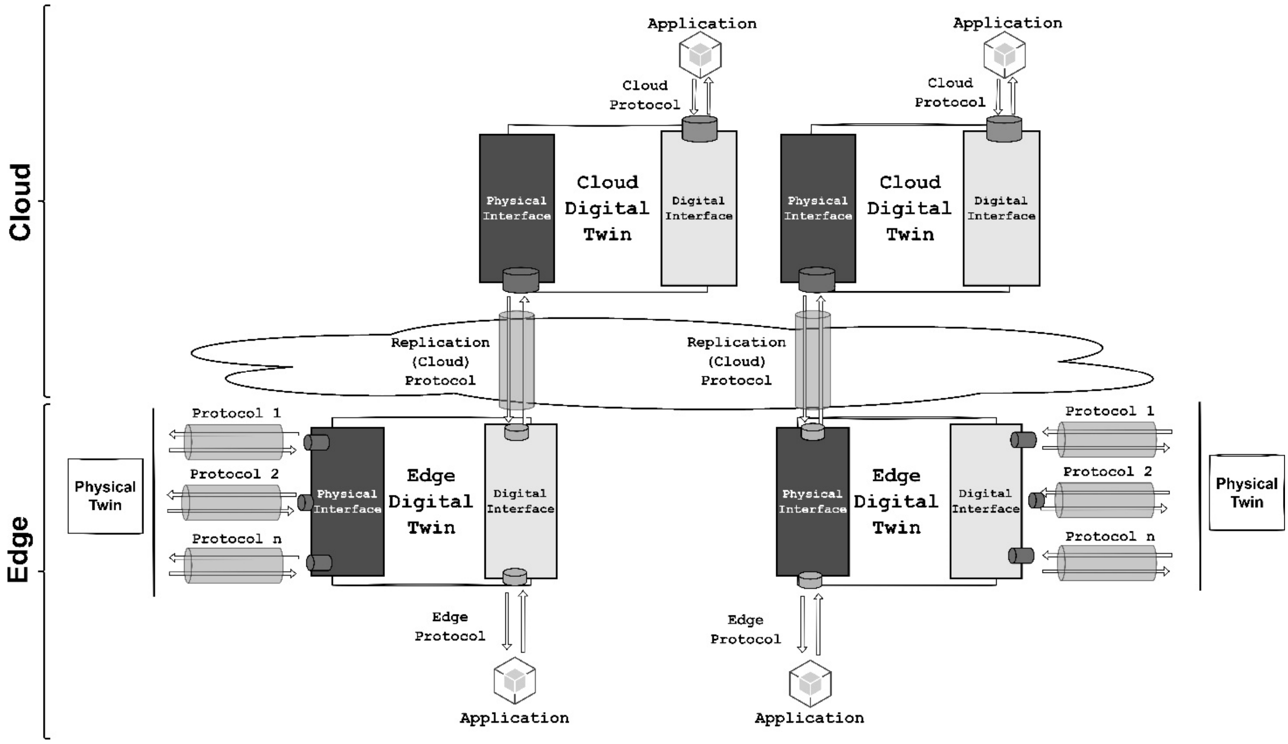

Figure 24: Relationships and opportunities of having DTs both on the Edge and in the Cloud

Edge deployments offer several advantages for DTs, particularly in scenarios where real-time processing and low latency are paramount. By placing computational resources closer to the physical assets, edge deployments minimize the delay between data generation and processing, enabling rapid response times and facilitating time-sensitive applications. This proximity also reduces the need to transmit large volumes of data to centralized servers, alleviating network congestion and bandwidth constraints. In this context, the following aspects should be considered with respect to the DT architectural components:

- **Physical Interface:** Edge deployments ensure that the PI directly interacts with the PTs, providing real-time data capture and an immediate actuation.
- **Model:** Local DTMs can be deployed on edge computational nodes to digitalized physical asset and analyse their behaviour in real time and keeping information on the local deployment and within the cyber-physical relationship between PTs and DTs.
- **State:** The states of the DT instances are maintained locally, allowing for quick updates and synchronization with the physical asset and a simplified and real-time access by the DI and external applications.
- **Storage:** DTs data is stored locally, reducing latency and ensuring immediate access to recent information and computed DTSs by edge applications and other DTs.
- **Digital Interface:** The DT allows for immediate integration with local systems and controls, facilitating quick decision-making and actions and higher level of local interoperability and contrast to physical layer fragmentation.

- **Management & Monitoring Interfaces:** The MOI and the MNI should oversee collecting and analysing performance metrics in real-time, ensuring timely detection and resolution of issues and to enable a local management of DT instances allowing dynamic reconfiguration and adaption of twin to the local variation of the operational context.

Edge deployments are also well-suited for environments with limited or intermittent connectivity, such as remote industrial sites or mobile assets. In these scenarios, DTs should be able to continue to operate autonomously, leveraging local data processing and storage capabilities until connectivity is restored.

On the other hand, cloud deployments offer scalability, accessibility, and centralized management capabilities. Cloud infrastructure provides virtually unlimited computational resources and storage capacity, allowing organizations to scale their DT deployments dynamically in response to changing demands. Cloud-based DTs also benefit from centralized management tools and services, streamlining deployment, monitoring, and maintenance processes. In this second configuration, the following aspects should be considered with respect to the DT architectural components:

- **Physical Interface:** The PI should be able to communicate with local devices and in some scenarios and deployment additional intermediary systems can be integrated in the architecture to aggregate data from physical assets before transmission to the cloud (DTs on the edge can embed this responsibility managing the complexity locally while interacting the cloud).

- **Model:** Advanced DTM should be executed in the cloud, utilizing high computational power for complex simulations and analytics exploiting the larger availability of data and information.

- **State:** The DTS of each DT running should be maintained centrally, ensuring consistency and coherence across multiple DT instances.

- **Storage:** Centralized storage systems in the cloud offer extensive capacity and robust data management for DTs.

- **Digital Interface:** The DI facilitates integration with other cloud-based services and applications, enabling the presentation of digital replicas to the cyber space and allowing advanced analytics and machine learning approaches on DTs data.

- **Management & Monitoring Interfaces:** Both the MOI and the MNI should provide a comprehensive overview of the system's health and performance associated to the running DTs, leveraging cloud-based analytics for deeper insights and providing an integrated management of cloud twin instances.

Moreover, cloud deployments enable seamless integration with other cloud-based services and analytics platforms, unlocking advanced capabilities such as machine learning, predictive analytics, and data visualization. By harnessing the power of the cloud, organizations can derive actionable insights from their DTs, driving operational efficiency, optimization, and innovation.

Hybrid deployments, which combine edge and cloud components, offer the best of both worlds. In a hybrid architecture, DTs can leverage edge computing for real-time processing and local decision-making, while also harnessing the scalability and advanced analytics capabilities of the cloud. This approach allows organizations to optimize resource utilization, minimize latency, and maximize flexibility, adapting to the specific requirements of each use case or environment. In this third configuration, the following aspects should be considered with respect to the DT architectural components:

- **Physical Interface:** Local interfaces at the edge should provide immediate data access for DTs toward the associated PTs, while cloud-based interfaces should be able to aggregate and analyse broader datasets and managing multiple entities at the same time on a larger scale.

- **Model:** Edge DTs and their DTMs handle real-time data, while cloud DTs and models perform deeper, more computationally intensive analyses such as training machine learning models or apply analytics and augmentation function on larger datasets.

- **State:** The DTS is managed locally on edge DTs for immediate operations with a target historical time frame while in the cloud DTSs are stored without any time limitation for an overall consistency and long-term tracking.

- **Storage:** DTs data should initially be stored at the edge for quick access and computation of digital replicas, then synchronized with cloud DT instances for long-term retention and analysis.

- **Digital Interface:** Hybrid interfaces allow seamless transition and integration between local and cloud systems and communication with local and remote distributed cyber applications.
- **Management & Monitoring Interfaces:** Local management and monitoring should ensure real-time performance assessment and configurability, while cloud functionalities should provide aggregated insights and trends analysis together with an integrated management solution for multiple DT instances.

Ultimately, the choice between edge and cloud deployments depends on factors such as latency requirements, data volume, connectivity constraints, and resource availability. By carefully evaluating these considerations and adopting a hybrid approach where appropriate, organizations can effectively deploy DTs through a shared and common architectural design to unlock their full potential in driving digital transformation and innovation across various industries and domains. The combination of edge and cloud capabilities ensures that DTs remain responsive, scalable, and adaptable to diverse operational environments.

6.3 Cyber-Physical Awareness

The effectiveness of DTs hinges on their ability to seamlessly bridge the cyber and physical realms. Cyber-physical awareness for DTs is the comprehensive understanding and live integration of digital models with their physical counterparts, achieved through continuous monitoring, data analysis, and feedback mechanisms. This concept embodies the entanglement of digital and physical realms, ensuring that the DT maintains the target fidelity in representing the state and behaviour of the physical asset throughout its life cycle. It involves the responsibility of accurately digitalizing a physical asset, capturing its dynamic characteristics, and enabling proactive management and decision-making to optimize performance, predict issues, and adapt to changing conditions. Cyber-physical awareness ensures that the DT evolves in sync with its physical counterpart, providing a reliable and actionable virtual representation that drives innovation and operational efficiency. This clause explores how DTs achieve this awareness by leveraging sensor data, IoT connectivity, and real-time feedback mechanisms, ensuring that they provide accurate and timely reflections of their physical counterparts through its architectural components and their functionalities. Cyber-physical awareness is essential for bridging the gap between the digital and physical domains. It involves the seamless integration of digital models with real-world physical assets, enabling DTs to accurately and in real-time reflect the state and behaviour of their physical counterparts. Understanding the intricacies of cyber-physical awareness is crucial for ensuring that Digital Twins provide accurate and timely reflections of their physical counterparts.

At the heart of cyber-physical awareness lies the ability of DTs to collect, process, and analyse data from their physical counterparts. This data is gathered through various sensors, IoT devices, and monitoring systems embedded within the physical assets. By continuously monitoring parameters such as temperature, pressure, vibration, and motion, DTs should be able to build a comprehensive understanding of the physical environment and its dynamics. In this context, the MNI plays a critical role in this process by configuring and managing the communication protocols and data acquisition settings of sensors and IoT devices that form the PI, ensuring seamless data collection from physical assets. The PI updates the DTM based on live physical data and incoming inputs, ensuring the digital representation through the DTS is accurate, and oversees data integrity and accessibility in the storage component. The DTS should be the element in charge of reflecting the current status and behaviour of the PT, ensuring that the data processed and handled by the DI is current and accurate, and uses the MOI to track performance metrics, data quality, and synchronization status between the physical and digital realms following also the evolution of DTs life cycles.

Collecting data is not sufficient; DTs should also interpret and respond to this data effectively. This requires robust communication channels and feedback mechanisms that enable bidirectional interaction between the digital and physical realms. For instance, if a sensor detects a deviation from the expected temperature range in a manufacturing plant, the corresponding DT should be able to analyse the data, identify potential causes or anomalies, and initiate appropriate actions or alerts. The MNI should facilitate real-time feedback mechanisms, ensuring immediate response to anomalies detected by sensors, adjusts the DTM configuration dynamically in response to real-time data, and reflects real-time changes and responses in the DTS component. It also communicates real-time alerts and adjustments to external systems and users through the DI and continuously monitors the impact of real-time adjustments and feedback on overall system performance via the MOI.

Cyber-physical awareness also encompasses monitoring and managing the quality of relationships between the DT and its physical counterpart. This includes factors such as network latency, packet loss, and data synchronization issues, which can significantly impact the accuracy and fidelity of the digital representation. Minimizing latency and ensuring reliable data transmission are critical for maintaining synchronization and providing timely insights. The MNI should ensures reliable and low-latency data transmission from physical sensors to the DT, monitors and adjusts the DTM based on the quality of incoming data to ensure high fidelity, and manages data synchronization and storage efficiency to support seamless data access and updates. It keeps the DTS component in sync with the physical asset's real-time status, addressing any discrepancies promptly, manages the data flow and communication protocols to maintain high responsiveness and reliability in the DI, and provides insights into network performance, latency issues, and synchronization status via the MOI, enabling proactive management.

To achieve effective cyber-physical awareness, DT architectures should be designed with careful consideration of the PI and DI involved. These interfaces should facilitate the collection of sensor data and environmental feedback from the physical world, while enabling communication, interaction, and feedback between the DT and external systems or users. By structuring these interfaces and integrating them seamlessly, DTs should be able to maintain an extended and updated knowledge of the quality of relationships with their physical counterparts, ensuring the fidelity and accuracy of their representations. Moreover, interoperability and standardization play a crucial role in ensuring consistency and compatibility across different DT implementations and deployments. By adopting shared architectures and standardized approaches to cyber-physical awareness, organizations can promote interoperability, facilitate knowledge sharing, and accelerate innovation in the rapidly evolving landscape of DT technology.

6.4 Digital Twin Monitoring & Management

The operational success of DTs should be based and rely on robust monitoring and management practices, which are crucial for ensuring their reliability, performance, and adaptability in dynamic environments. Monitoring DTs should involves tracking various metrics and indicators to assess their performance, resource utilization, and adherence to predefined objectives also through the evolution of their lifecycle. This includes monitoring parameters such as CPU and memory usage, network latency, response times, error rates, data consistency, entanglement level, fidelity and life cycle phases. Continuous monitoring allows stakeholders to identify potential issues, anomalies, or performance bottlenecks and take proactive measures to address them.

Several monitoring techniques and tools are available and can be applied within the DT ecosystem, ranging from traditional monitoring solutions to more advanced analytics and visualization platforms. The integration of these tools with DTs should offer and enable live and real-time dashboards and reporting tools, alerts, and reports that provide insights into the health and status of twins, enabling stakeholders to make informed decisions and optimizations.

EXAMPLE: Time-series databases and monitoring platforms like Prometheus and Grafana enable the collection, storage, and visualization of time-series data, allowing stakeholders to analyse trends and patterns over time can be integrated with the MOI of a DT instance enabling an integrated and interoperable data collection.

At the same time, an effective MNI on a DT should be equally essential for configuring, updating, and managing DTs throughout their life cycle. The MNI should provide administrators with the ability to perform tasks such as configuring twins instances, updating software versions or sum-modules, configuring parameters, and scaling resources based on changing requirements. These interfaces should be intuitive, user-friendly, and capable of handling complex operations seamlessly. Various management techniques and tools can be adopted for DTs, depending on the specific requirements and architecture of the system. Container orchestration platforms like Kubernetes can provide powerful tools for deploying, managing, and scaling containerized applications and the integration with DTs should enable a more integrated deployment approach and a cloud-native solution for twin architecture design, implementation and execution. Configuration management tools such as Ansible and Puppet allow automate the process of configuring and provisioning infrastructure, ensuring consistency and reliability across deployments. Additionally, version control systems like Git enable collaboration and versioning of DT configurations and codebases, facilitating reproducibility and traceability.

The MNI should play a pivotal role in this ecosystem by providing a structured and interoperable point of control and coordination for DTs activities event through different architectures and implementations. It should allow the communications with various architectural components of the DT. With the PI, the MNI manages the integration and interaction of physical assets, ensuring accurate data flow from sensors and actuators. It should also interact with the DTM by updating and refining the digital representation based on live data and contextual information. The MNI also oversees the Storage component, ensuring efficient data management, retrieval, and archival processes. In relation to the State, the MNI should ensure that the updated and fresh state of the DT is accurately maintained and synchronized with its physical counterpart. The DI, should be responsible for external communications and interactions, is configured and managed by the MNI to facilitate seamless data exchange and interoperability with other systems. Lastly, the MNI should work closely with the MOI to track performance, health, and behaviour, enabling proactive maintenance and optimization of DTs.

In summary, effective monitoring and management should be essential components of DT architectures, enabling stakeholders to ensure the reliability, performance, and adaptability of their implementations. By leveraging monitoring tools and management interfaces, organizations can gain valuable insights into the behaviour and performance of DTs, optimize their configurations, and respond promptly to changing requirements and operational conditions.

6.5 Digital Twin Discoverability

Discovering and accessing DTs instances within a complex ecosystem is crucial for their applicability considering both a local discoverability and a distributed on with respect to the vision of building an ecosystem of DTs as illustrated in Figure 25. Robust discoverability capabilities as supported by [i.7] and [i.12] should facilitate seamless interaction and collaboration within digital ecosystems among DTs, applications, and users. Discoverability involves the ability to locate, access, and understand DTs, their descriptions, and their states, both internally within the DT architecture and externally for applications, services, and digital entities.

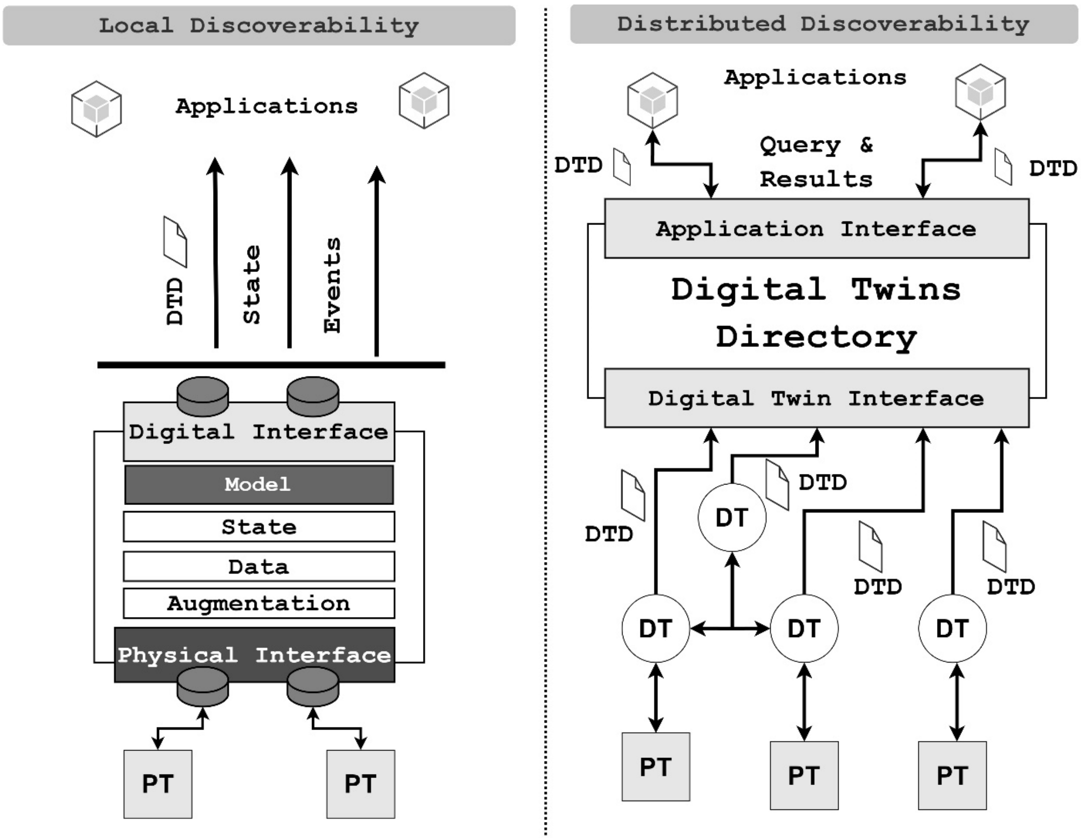

Figure 25: Schematic representation of DT discoverability opportunities and capabilities

Internally, DTs should enhance discoverability through their DI with its digital adapters, and through the DTD. These components provide structured information about the DT's identity, capabilities, and interfaces, enabling stakeholders to identify and interact with them effectively. By leveraging standardized formats and protocols, such as RESTful APIs or queries languages, DTs should be able to offer rich discoverability features that support efficient search and querying. The DI should communicate with these components, ensuring that the necessary metadata and descriptors are available for discoverability. The DI coordinates the information flow, updating the DTDs and ensuring the digital adapters are properly configured for effective discovery.

Externally, discoverability extends to broader architectural components that support the discovery of DTs across a distributed ecosystem where multiple twins can be executed even through different implementations and platforms. In this context, a DT repositories or inventory should support the architecture serving as centralized collection of DTDs, and potentially also maintaining snapshots of the latest DTSs, enabling stakeholders to search, query, and discover relevant DTs based on various criteria. These repositories should expose discovery APIs or interfaces that support structured search and filtering capabilities, allowing stakeholders to locate twin based on their physical attributes, properties, or relationships and with respect to the PTs that they are representing. The DI should play a pivotal role in managing the interactions with these repositories, ensuring that the DT metadata is up-to-date and accurately reflects the current state and capabilities of the twins.

Moreover, discoverability capabilities should enable search and discovery based on PT information, such as location, type, or context. This integration of PT attributes into discoverability mechanisms enhances the relevance and accuracy of search results, ensuring that stakeholders can find DTs that closely align with their needs and requirements. The PI provides the necessary data about the PT, which the DTM and the DI uses to update the DT repositories and enhance search information and accuracy.

Incorporating discoverability features into DT architectures fosters collaboration, interoperability, and innovation within digital ecosystems. By providing stakeholders with efficient means to locate and engage with DTs, discoverability facilitates seamless integration and interaction across diverse applications, services, and domains, ultimately driving value and impact in the digital landscape.

Annex A (informative): Change History

Date	Version	Information about changes
September 2023	V0.0.1	Early Draft
December 2023	V0.0.2	Interim Draft
March 2024	V0.0.3	Stable Draft
May 2024	V0.0.4	Final Draft
July 2024	V1.1.1	Final Review by Technical Officer before ***editHelp!*** publication pre-processing
August 2024	V1.1.1	First published version

History

Document history		
V1.1.1	August 2024	Publication