

Data Space for Digital Twin applications - Functional Architecture - Release 2

Classification: APPROVED Reference Document

Confidentiality: PUBLIC

Version: 2

DSDT FA rel2

July 2026

Legal

THIS DOCUMENT HAS BEEN DESIGNATED BY THE INNOVATIVE OPTICAL AND WIRELESS NETWORK GLOBAL FORUM, INC. (“IOWN GLOBAL FORUM”) AS AN APPROVED REFERENCE DOCUMENT AS SUCH TERM IS USED IN THE IOWN GLOBAL FORUM INTELLECTUAL PROPERTY RIGHTS POLICY (THIS “REFERENCE DOCUMENT”).

THIS REFERENCE DOCUMENT IS PROVIDED “AS IS” WITH NO WARRANTIES WHATSOEVER, WHETHER EXPRESS, IMPLIED, STATUTORY, OR OTHERWISE, INCLUDING WITHOUT LIMITATION ANY WARRANTY OF MERCHANTABILITY, NONINFRINGEMENT OF THIRD PARTY RIGHTS, TITLE, VALIDITY OF RIGHTS IN, FITNESS FOR ANY PARTICULAR PURPOSE, OR ANY WARRANTY OTHERWISE ARISING OUT OF ANY PROPOSAL, REFERENCE DOCUMENT, SAMPLE, OR LAW. WITHOUT LIMITATION, IOWN GLOBAL FORUM DISCLAIMS ALL LIABILITY, INCLUDING WITHOUT LIMITATION LIABILITY FOR INFRINGEMENT OF ANY PROPRIETARY RIGHTS AND PRODUCTS LIABILITY, RELATING TO USE OF THE INFORMATION IN THIS REFERENCE DOCUMENT AND TO ANY USE OF THIS REFERENCE DOCUMENT IN CONNECTION WITH THE DEVELOPMENT OF ANY PRODUCT OR SERVICE, AND IOWN GLOBAL FORUM DISCLAIMS ALL LIABILITY FOR COST OF PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES, LOST PROFITS, LOSS OF USE, LOSS OF DATA OR ANY INCIDENTAL, CONSEQUENTIAL, DIRECT, INDIRECT, PUNITIVE, EXEMPLARY, OR SPECIAL DAMAGES, WHETHER UNDER CONTRACT, TORT, WARRANTY OR OTHERWISE, ARISING IN ANY WAY OUT OF USE OR RELIANCE UPON THIS REFERENCE DOCUMENT OR ANY INFORMATION HEREIN.

EXCEPT AS EXPRESSLY SET FORTH IN THE PARAGRAPH DIRECTLY BELOW, NO LICENSE IS GRANTED HEREIN, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE, TO ANY INTELLECTUAL PROPERTY RIGHTS OF THE IOWN GLOBAL FORUM, ANY IOWN GLOBAL FORUM MEMBER OR ANY AFFILIATE OF ANY IOWN GLOBAL FORUM MEMBER. EXCEPT AS EXPRESSLY SET FORTH IN THE PARAGRAPH DIRECTLY BELOW, ALL RIGHTS IN THIS REFERENCE DOCUMENT ARE RESERVED.

A limited, non-exclusive, non-transferable, non-assignable, non-sublicensable license is hereby granted by IOWN Global Forum to you to copy, reproduce, and use this Reference Document for internal use only. You must retain this page and all proprietary rights notices in all copies you make of this Reference Document under this license grant.

THIS DOCUMENT IS AN APPROVED REFERENCE DOCUMENT AND IS SUBJECT TO THE REFERENCE DOCUMENT LICENSING COMMITMENTS OF THE MEMBERS OF THE IOWN GLOBAL FORUM PURSUANT TO THE IOWN GLOBAL FORUM INTELLECTUAL PROPERTY RIGHTS POLICY. A COPY OF THE IOWN GLOBAL FORUM INTELLECTUAL PROPERTY RIGHTS POLICY CAN BE OBTAINED BY COMPLETING THE FORM AT: www.iowngf.org/join-forum. USE OF THIS REFERENCE DOCUMENT IS SUBJECT TO THE LIMITED INTERNAL-USE ONLY LICENSE GRANTED ABOVE. IF YOU WOULD LIKE TO REQUEST A COPYRIGHT LICENSE THAT IS DIFFERENT FROM THE ONE GRANTED ABOVE (SUCH AS, BUT NOT LIMITED TO, A LICENSE TO TRANSLATE THIS REFERENCE DOCUMENT INTO ANOTHER LANGUAGE), PLEASE CONTACT US BY COMPLETING THE FORM AT: <https://iowngf.org/contact-us/>]

Copyright © 2026 Innovative Optical Wireless Network Global Forum, Inc. All rights reserved. Except for the limited internal-use only license set forth above, copying or other forms of reproduction and/or distribution of this Reference Document are strictly prohibited.

The IOWN GLOBAL FORUM mark and IOWN GLOBAL FORUM & Design logo are trademarks of Innovative Optical and Wireless Network Global Forum, Inc. in the United States and other countries. Unauthorized use is strictly prohibited. IOWN is a registered and unregistered trademark of Nippon Telegraph and Telephone Corporation in the United States, Japan, and other countries. Other names and brands appearing in this document may be claimed as the property of others.

Contents

1. Introduction	5
1.1. Scope	5
1.2. Structure of this document	6
1.3. Update from Release 1 of DSDT Functional Architecture	6
2. Concept	8
3. Background and Rationale	10
3.1. Vision and technical requirements	10
3.2. Business Added Values	12
4. Functional Architecture	13
4.1. Overview	14
4.2. Stakeholder Patterns	15
4.3. Functions	16
5. Security Considerations	40
6. Interaction with IOWN infrastructure	41
6.1. Interaction with IDH	41
6.2. Interaction with DCI systems	42
6.3. Interaction with APN	44
7. Interaction with external systems and users	45
8. Conclusion	47
Appendix. Example instances	48
References	53
Abbreviations and acronyms	55
History	57

List of Figures

Figure 1: Digital Twin concept	8
Figure 2: Digital Twin concept and identified functions.	13
Figure 3: Data Space for Digital Twin applications (DSDT) functional architecture overview.	14
Figure 4: Fundamental interaction patterns for Digital Twin applications.	16
Figure 5: Function elements of the registry function	18
Figure 6: Workflow for the registration of Digital Twin components	18
Figure 7: Function elements of the federation function	19
Figure 8: Federated Database (FDB) workflow	20
Figure 9: Translation between two digital twins.	21

Figure 10: Translation function workflow.	21
Figure 11: Function elements of the brokering function.	22
Figure 12: Workflow of the broker function.	22
Figure 13: Elements of the Synchronization function	23
Figure 14: Synchronization function workflow.	24
Figure 15: Actuation function overview.	25
Figure 16: Digital Twin orchestration function operations.	26
Figure 17: Digital Twin orchestration function interface overview	27
Figure 18: Analytics monitoring function interface overview.	30
Figure 19: Cross-domain analytics interface overview	31
Figure 20: Example of Probabilistic Digital Twin Data Structure	33
Figure 21: Functional block patterns of the Probabilistic Digital Twin function	34
Figure 22: Simulation function overview	36
Figure 23 Visualization function overview.	36
Figure 24: Digital Twin manipulation function interface overview.	37
Figure 25: Exposed interfaces of the overall DSDT layer. This figures summarizes all the exposed interfaces of all the DSDT functions.	45

1. Introduction

Digital Twin [IOWN-GF-DTC] conceptualizes the approach to re-create in the digital world a replica of an asset in the real world for easing the management of such real asset. The Digital Twin encompasses multiple and heterogeneous technologies that spans the data management, the data computation, analytics and visualization. Further, a real asset can be relevant for different stakeholders each holding a view of it. Therefore, sharing information of the Digital Twin among stakeholders is necessary. A **Data Space** [IOWN-GF-Vision-2030] allows the connection of multiple data sources and providers regardless of the level of integration towards a data economy. Data Space technologies are fundamental to facilitate Digital Twin implementation especially in complex use cases. The realization of Digital Twin that is highly accurate to its related real asset requires large amount of resources in terms of data storage, computing powers, and communication. IOWN infrastructure offers such capabilities through the exploitation of all-photonic network communication that boosts the data transmission while significantly reducing latency and energy consumption [IOWN-GF-Vision-2030].

For this purpose, this document specifies a Functional Architecture of the Data Space for Digital Twin applications (DSDT FA) layer to enable and facilitate the realization of applications using Digital Twin technologies in a Data Space [IOWN-GF-Vision-2030] setting and leveraging the IOWN infrastructure capabilities.

This document represents the second release of the DSDT Functional Architecture and supersedes Release 1 [IOWN-GF-DSDT-FA-rel1]. This functional architecture enhances the previous set of functionalities across data handling, digital twin management, analytics, and manipulation by introducing extended features and new capabilities. In addition, it introduces a new category of functions dedicated to digital twin security. These enhancements aim to address a broader set of requirements, particularly to support the development of the Network Digital Twin (NDT) as a first concrete implementation of digital twin use cases.

1.1. Scope

This document defines the functional architecture of the Data Space for Digital Twin applications (DSDT) layer at the top of the IOWN technical infrastructure. DSDT encompasses functional blocks that enable the implementation of the digital twin applications allowing and easing the exploitation of the IOWN technical infrastructure. The functions spans the data management and integration for the Digital Twin, the support for the data analytics to model the real asset behaviour in the digital world, and the interaction with user through manipulation of the Digital Twin.

1.2. Structure of this document

The DSDT FA in this document presents logical functions. These functions have very specific and scoped behaviour and interact with other functions to build up the management of Digital Twins. Different concrete implementation of the DSDT might combine different set of the DSDT FA functions for different purpose.

The structure of this document offers different perspectives to the reader:

- Digital Twin concept: Section 2 presents the Digital Twin concept consistent with the DSDT FA and with IOWN user case documents [IOWN-GF-DTF-AR; IOWN-GF-DTF-NDT].
- Digital Twin requirements: Section 3 presents the requirements identified for IOWN use cases and to be addressed.
- Digital Twin functional architecture: Section 4 presents the functional architecture of DSDT FA from an overview perspectives explaining the functions blocks till detailed explanation of every function
- IOWN infrastructure manager: Section 5 presents the functions that DSDT exploits from other IOWN technologies
- Interaction with external system and users: Section 6 summarizes the interfaces used by external systems to interact with the DSDT such as inputting data sources, interact with the Digital Twin for application or visualization, develop the Digital Twin by inputting components (such as operators), and actuate decisions on real assets.

1.3. Update from Release 1 of DSDT Functional Architecture

This new release of the Data Space for Digital Twin applications is more mature than the previous release [IOWN-GF-DSDT-FA-rel1] since it is the result of direct experience of digital twin system design such Network Digital Twin use case [IOWN-GF-NDT-REQ]. In particular the changes mapping are as follow:

- Function Translation (section 4.2.1) has evolved to address UC1-REQ-DT-DATA-MODEL-02
- Function Actuation (section 4.2.1) has evolved to address UC1-REQ-DT-ACT-DEVICE-03
- Function Analytics orchestration (section 4.2.2) has evolved to address UC1-REQ-DT-M-PROC-01
- A new function Analytics monitoring (section 4.2.2) has been added to address UC1-REQ-DT-M-PROC-[02,03,05,06],
- Visualization workflow (section 4.2.3) has evolved to address UC1-REQ-DT-M-USER-01 and UC1-REQ-DT-M-USER-03
- A whole new section has been added address security functions and in particular:
 - Data Access control function to address UC1-REQ-DT-DATA-SEC-[1,2]

- Data Usage Control Orchestration to address UC1-REQ-DT-DATA-SEC-6
- Attestation function, tracing/auditing function to address UC1-REQ-DT-DATA-SEC-3
- Monitoring for Early Detection against Unknown Attacks to address UC1-REQ-DT-DATA-SEC-[8,9]
- Further security considerations are also extended and in particular:
 - Post-quantum security considerations to address UC1-REQ-DT-DATA-SEC-5
 - Resilience (Response to Unknown Attacks) to address UC1-REQ-DT-DATA-SEC-[7,8,9]

2. Concept

The Digital Twin has not a unique definition and it is described and adapted in different contexts. The general concept of the Digital Twin taken into consideration in this document follows the one described in the IOWN Network Digital Twin use case document [IOWN-GF-NDT] and IOWN Digital Twin Framework Analysis Report [IOWN-GF-DTF-AR] : a Digital Twin is a digital representation of a real asset and it reflects the status of the real asset at a certain fidelity (depending on the application use case) by integrating static, semi-static, real-time data, and analytics functions that augment the raw data with inference and reasoning. In addition the Digital Twin encompasses capabilities that allows to replicate, simulate and analyze behaviour in the digital world without interfering with the real asset. Further, some capabilities allows to take decision to be actuate on the real asset to achieve a target state.

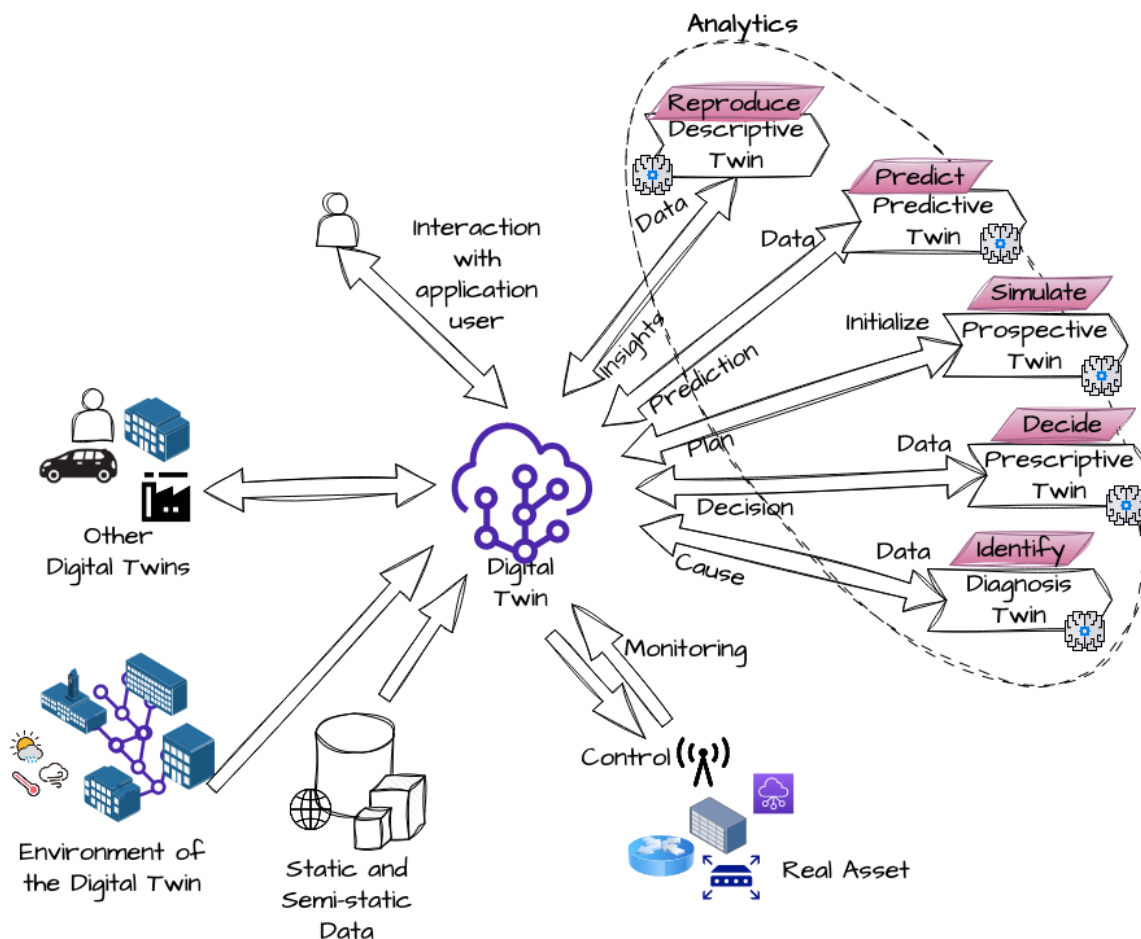


Figure 1: Digital Twin concept

Figure 1 depicts the concept of the Digital Twin with its different components. Heterogeneous data sources are combined together converging in a single representation of the real asset (or part of it). The data is combined in a digital model that is a overlay data structure that links together all the available data. Data sources are, but not limited to:

- **Static and semi-static** data such a real asset description and specifications.
- **Monitored status** of the real asset such as through an IoT deployment or other real-time data.
- Information about the **environment** of the digital twin such as spatial information or weather forecasts.
- **Other digital twins** of heterogeneous types.

The data is, then, processed by analytics for augmenting the combined data and recreate the behaviour of the real asset. Also the analytics are integrated and linked in the digital model, thus, the Digital Twin is composed by both the data and the analytics functions that recreate its state and behaviour.

ETSI ISG CIM [ETSI CIM017] defines different analytics capabilities for the Digital Twin such as follows:

- *Descriptive Twin* represents the status of the digital twin including observations and inferred insights not directly observed through monitoring (e.g., anomaly detection or pattern recognition) from different stakeholders and administrative domains
- *Predictive Twin* that predicts future status of the real asset given the current status and the past recorded behaviours (e.g., foreseen misbehaviour)
- *Prospective Twin* to simulate hypothetical (“what-if”) scenarios given the current understanding of the real asset comprising of the descriptive (monitoring and insight inference) and predictive modules.
- *Prescriptive Twin* that includes functions to decide upon actions aiming at controlling the real asset to follow a target state.
- *Diagnosis Twin* that allows to understand the causes of a detected situation of the real asset.

The mentioned analytics are leveraging on a holistic approach for better performance in the results.

The data and the analytics of the Digital Twin are distributed among different stakeholders in a **data space**. Each stakeholder might own and manage a different view of the same digital twin that is combined in the data space respecting the collaboration agreements between the stakeholders. Functions to negotiate and instantiate agreements between parties are outside the scope of the IOWN DSDT function architecture and supposed to be provided by other technologies.

3. Background and Rationale

3.1. Vision and technical requirements

The IOWN Vision 2030 [IOWN-GF-Vision-2030] document introduces the *Data Space* vision where “data is collected from different data sources coming from very different domains”. In addition, the data space requires *Data Sovereignty* that “aims at addressing the concerns coming from companies on the impossibility to share their data with other companies, because there is a risk that important data and business secrets could be exposed”.

Digital Twin Framework analysis report [IOWN-GF-DTF-AR] studied IOWN-enabled use cases that embodies the IOWN Vision 2030. The analysis results with identification of requirements and gaps for their realization. The results of the DTF analysis report identifies the following requirements:

- *DTF-Req-1*: The **IOWN GF infrastructure**, including APN and DCI, is required to be **flexible** to accommodate various types of use cases that have critical differences of static and dynamic composition of digital twins.
- *DTF-Req-2*: The DTF needs to find a way to define **static and dynamic composition of digital twins** based on use case scenarios and preconditions to identify concrete requirements for the IOWN GF infrastructures.
- *DTF-Req-3*: The IOWN GF infrastructure needs to provide a platform to **manage data flows of digital twins between different stakeholders**. This requires access right management between stakeholders in terms of confidentiality, privacy, and efficiency of data exchange.
- *DTF-Req-4*: The IOWN GF needs to provide a mechanism to manage **confidentiality and privacy of different parts of one digital twin**. This could be a guideline for structure of a digital twin with explicit indication of confidentiality and could need to collaborate with other SDOs.
- *DTF-Req-5*: The IOWN GF needs to identify an **interoperability mechanism** to interact different types of digital twins. This could require collaboration with other SDOs.
- *DTF-Req-6*: The IOWN GF needs to provide a **data usage control system** to control how the data is used after it is accessed by a data consumer.

In addition to the previously described requirements for a high-level digital twin in the data space, this release also targets a specific use case category: the Network Digital Twin (NDT), as a first concrete implementation example. Release 2 of the DSDT Functional Architecture addresses the general requirements needed to enable NDT implementations, while leaving the detailed interfaces for network duplication and management to specific implementations. The following provides a brief summary of these requirements (the full set is available in [IOWN-GF-

NDT-REQ]; the specific NDT interfaces not considered part of the DSDT FA release 2, are marked in italic a):

- **Data Model Requirements:** a network digital twin must support *YANG*-based data models for interoperability and allow new models to be added without breaking existing ones. It should represent complex, multi-layer networks using scalable multigraph/property graph approaches (e.g., NGS-LD, OWL) and provide APIs for integrating and exporting data. Compatibility with existing network management systems and technologies is also expected.
- **Network Interface Requirements:** the digital twin must interact with the real network via the DT-act-device interface (part of the DSDT FA since release 1 [IOWN-GF-DSDT-FA-rel1]) to initialize, synchronize, and control the network through the *APN* controller. It must support Transport *PCE/TAPI* APIs, enable configurable and incremental updates, and integrate network telemetry (e.g., via *gRPC* and *gNMI*) to keep the twin accurate and enriched with real-time and aggregated data.
- **Data Storage Requirements:** the digital twin must use a property graph-based storage system to manage network data, supporting CRUD operations, graph queries, and retrieval of both individual resources and full/subgraph views. It should also handle historical data to enable analysis and predictions, while providing APIs for data export (e.g., JSON, RDF) and optional support for geo-spatial information.
- **Processing Requirements:** the digital twin must support simulation, analytics, and optimization tasks to assist network management and decision-making. It should provide APIs to manage and monitor simulation processes, generate and retrieve results, and optionally export them. The system should also support advanced computations (e.g., AI, optimization) and may leverage scalable computing resources and external tools to enable predictive analysis and network optimization.
- **External API Requirements:** the digital twin must expose developer-friendly interfaces, primarily through REST APIs, to enable applications like network visualization and decision-making tools. It should support common data formats such as JSON and RDF-based syntaxes (e.g., JSON-LD, Turtle), ensuring easy integration and development of user-facing and shared software components.
- **Data Security Requirements:** the digital twin must protect the confidentiality, integrity, and availability of its data, especially for sensitive or critical information. Data should be classified by security level, with strict access control for confidential data and protection against unauthorized modifications. The system should also support identifying and monitoring critical data, detecting tampering or theft, and may include advanced features like auditability, data sovereignty, and resilience against emerging threats (e.g., quantum attacks)

The DSDT functional architecture aims at covering these gaps by facilitating the development of Digital Twin applications through a Data Space powered by IOWN infrastructure.

3.2. Business Added Values

A Data Space transforms fragmented, siloed data into a shared, trusted, interoperable, and governed ecosystem that unlocks innovation, efficiency, and sustainable business models for smart solutions. It enables secure and sovereign data sharing among multiple stakeholders, ensures interoperability across heterogeneous sources (IoT, legacy systems, static data), and fosters data value creation by combining cross-domain datasets to support new models such as data-as-a-service and pay-per-use. By providing a scalable, reusable infrastructure, it replaces one-off integrations with replicable smart solutions while strengthening collaboration and cross-domain ecosystem building. Complementing this, a Digital Twin converts static and real-time data into actionable, dynamic intelligence through holistic 360° monitoring, cross-domain integration, simulation and predictive analytics, and context-aware maintenance that improves reliability and sustainability. Together, Data Spaces act as the trusted data-sharing backbone while Digital Twins serve as the actionable intelligence engine, forming a closed value loop—data → simulation → insights → new data → innovation—where the Data Space feeds high-quality governed data to the Twin, and the Twin enriches the ecosystem with predictions and derived insights, reinforcing trust, collaboration, and continuous improvement.

4. Functional Architecture

The Data Space for Digital Twin applications (DSDT) layer addresses requirements identified in section 3 [IOWN-GF-DTF]. The functional architecture of the DSDT comprises several functions to enable the implementation of Digital Twin applications through the Data Space leveraging the technologies from IOWN. The figure below maps the identified functions in the Digital Twin concept presented in Section 2.

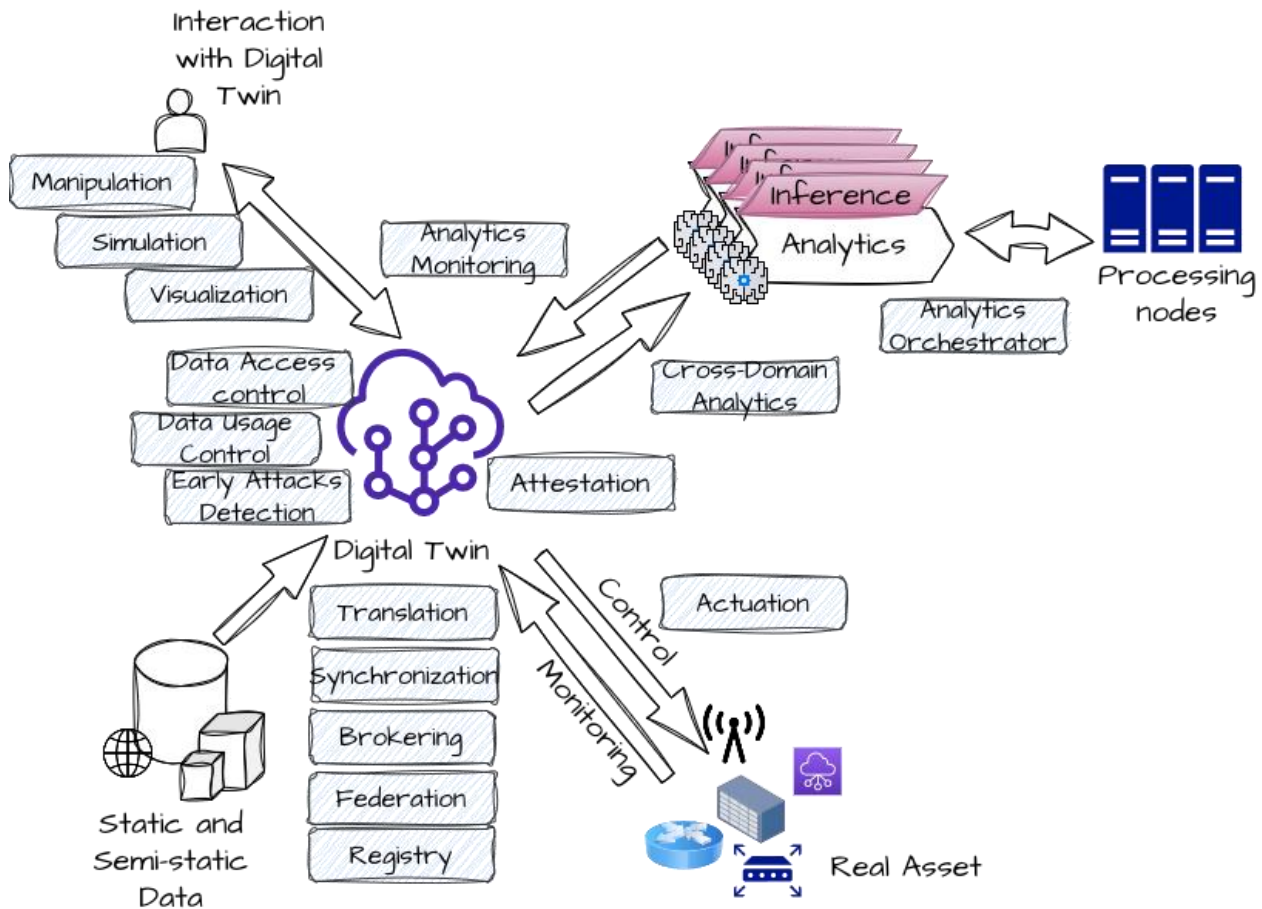


Figure 2: Digital Twin concept and identified functions.

The Digital Twin at the center of the above image is not a single entity in a single server, but is a concept shared among multiple data and processing nodes in multiple forms (heterogeneous data sources and heterogeneous analytics). The functions are intended to manage the Digital Twin for different purpose such as data integration and handling, analysis and augmentation, and interaction with the user. The following sections are explaining these functions and their interactions in more details.

4.1. Overview

The Figure below shows the overview of the Data Space for Digital Twin applications (DSDT) layer. The DSDT layer facilitates the implementation and execution of Digital Twin application in a Data Space environment where multiple stakeholders collaborate into a federation.

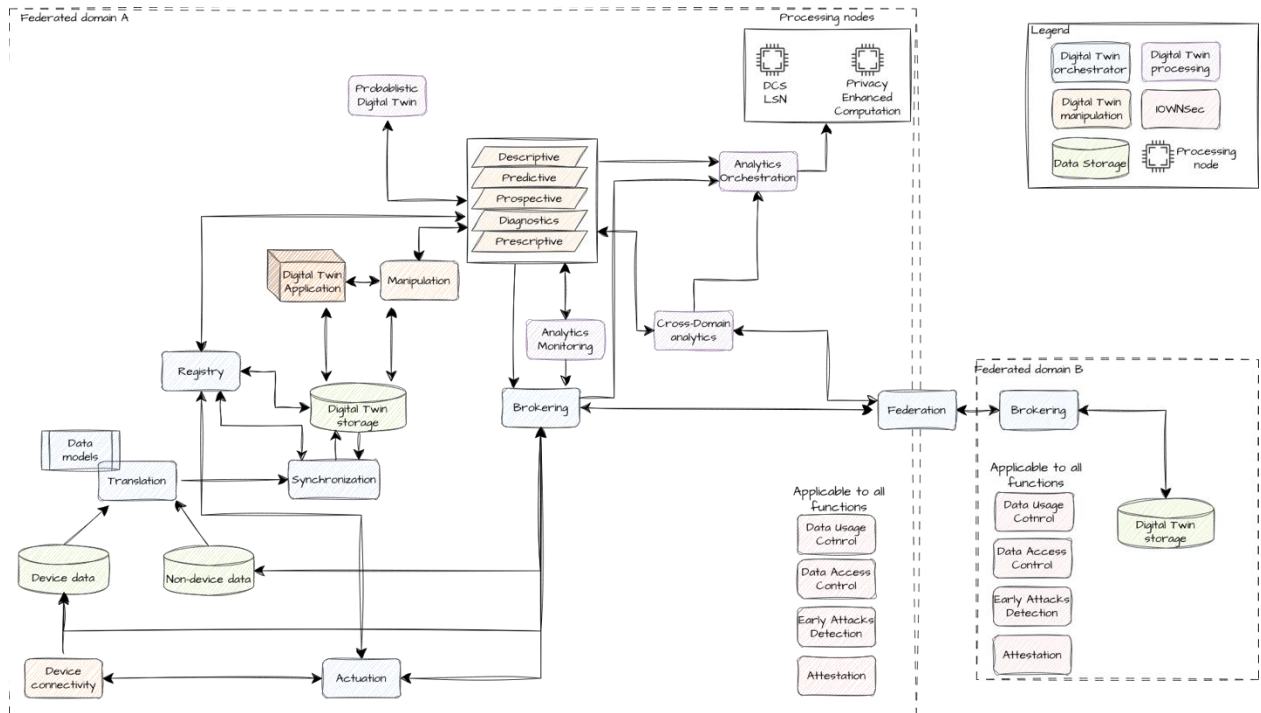


Figure 3: Data Space for Digital Twin applications (DSDT) functional architecture overview.

The functional overview for enabling the DSDT can be grouped in the following:

- **Digital Twin data handling** targets information sharing and interaction among heterogeneous stakeholders, each of them managing a different version of the same digital twin or other digital twins.
- **Digital Twin analytics** comprises functions to handles data analytics services in the data space across multiple stakeholders and in distributed and disaggregated processing nodes.
- **Digital Twin manipulation** comprises functions that enables the user to interact with the Digital Twin. Simulation and visualization offer opportunity to the users to have immediate view of the digital twin, manipulate it and see the effects, thus actuating the real world as-sets.
- **Digital Twin security** comprises functions for the data access control, data usage control orchestration, configuration attestation and early detection of unknown attacks.

In addition the Digital Twin will use additional functions offered by other components of the IOWN GF technology infrastructure, such as:

- Security that enables the interaction in the data space of multiple stakeholders seamlessly addressing the security threats.
- Data Storage handles different heterogeneous parts of the digital twin data. This part is provided by IOWN Data Hub (IDH) [IOWN-GF-IDH].
- Processing capabilities permits the data analytics to be executed. This part is provided by IOWN Data-Centric Infrastructure (DCI) [IOWN-GF-DCI] and by the Privacy-Enhancing computing Technologies (PET).

This usage of these additional functionalities are described in section 5.

4.2. Stakeholder Patterns

The DSDT functional architecture enables the data consumption over distributed data servers and across multiple stakeholders. To simplify the discussion of the security requirements and the usage of distributed data storage, this section defines fundamental patterns that can be combined to cover every case of Digital Twin applications.

The patterns are presented in the picture below and consist of:

- Single Stakeholder data consumption
 - Single server: a single stakeholder provides and consumes data all in a centralized server instance.
 - Multiple server: a single stakeholder provides and consumes data. The data consuming service and the data storage are in two different computing nodes.
 - Distributed storage: a single stakeholder provides and consumes data. The data is distributed among 2 or more data nodes. The data consuming server is placed in yet another processing node.
- Multiple Stakeholders (data space) data consumption
 - Simple: This is the basic interaction between one stakeholder that provides data and one stakeholder that consumes data. The data consuming service and the data storage are running into different processing nodes. In some cases, the physical processing node might be the same but logically (due to isolation techniques) the two processing nodes are separated.
 - Multiple data sources: This is the interaction between multiple stakeholders in the data space where multiple data producers supply data to a data consuming process

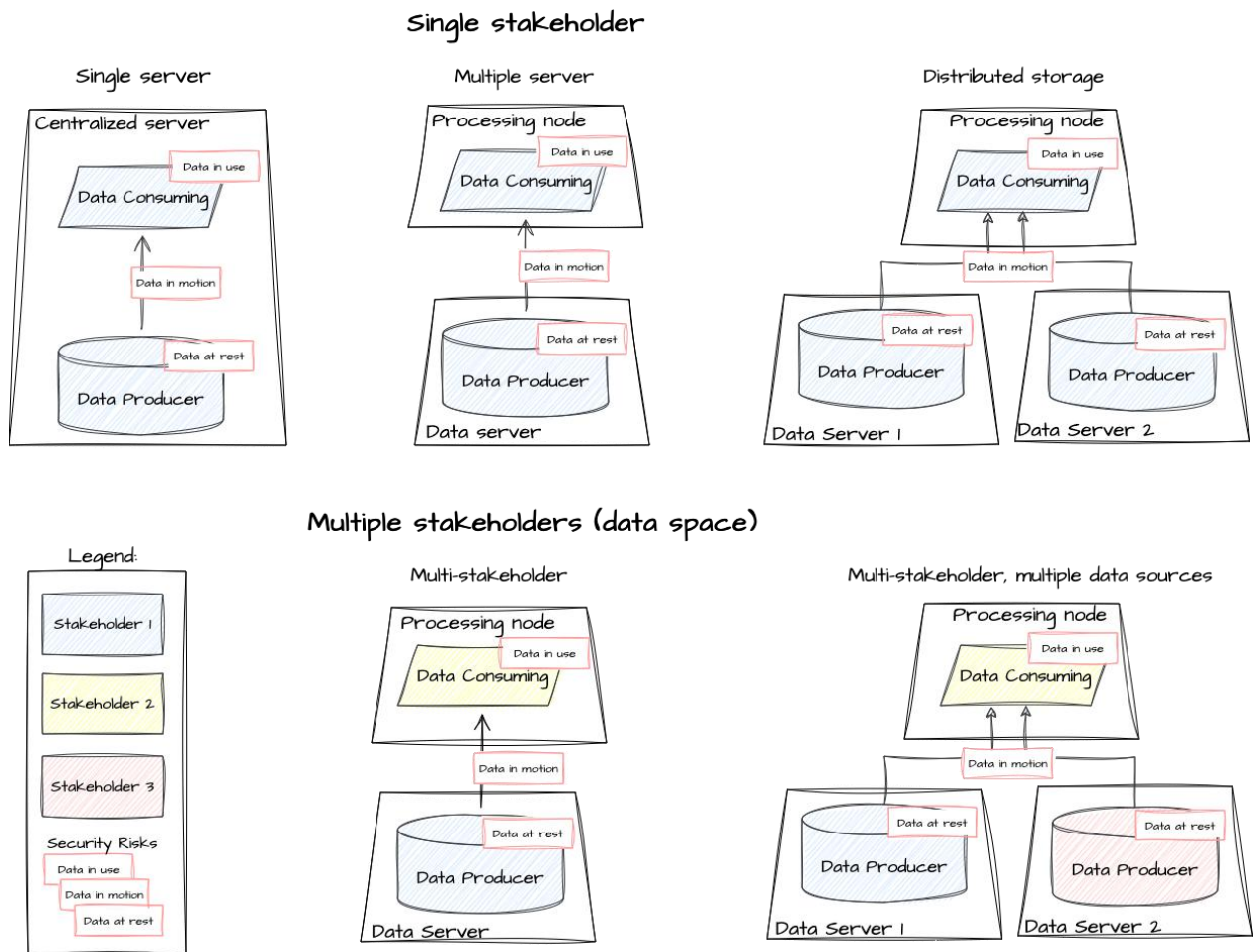


Figure 4: Fundamental interaction patterns for Digital Twin applications.

4.3. Functions

4.3.1. Digital Twin data handling functions

Digital Twin data handling functions enable communication and synchronization within digital twin systems. Initially, communication occurs between physical entities and their digital representations, as well as between different digital twins. This communication transitions from synchronizing individual objects to sharing information among groups of objects with complex interactions, spanning multiple segments or domains. Synchronization plays a crucial role in ensuring that data and actions are aligned in real-time across these interconnected systems, allowing for accurate and cohesive operations. For instance, urban digital twins enable cities to collect data through IoT sensors for decision-making and problem-solving in areas like urban planning, environmental management, and energy control. The functions that handle the data of Digital Twin consists of: Registry, Federation, Translation, Brokering, Synchronization and Digital Twin Linking.

- The **Registry** registers and discovers digital twin components based on their feature information.

- The **Federation** configures and manages a federation of digital twins that updates a shared virtual model while maintaining the privacy data generated by physical objects and autonomy processes in individual digital twins.
- The **Brokering** identifies and authenticates the digital twin, relays data transmission and reception, and performs data filtering, real-time delivery, and guaranteed delivery.
- The **Synchronization** synchronizes many-to-many interactions between physical space entities and cyberspace models between digital twins.
- The **Translation** facilitates formal and semantic transformation of communications between digital twins in different domains.
- The **Actuation** function ensures synchronization between a digital twin and its real-world asset by enforcing changes within the asset's capabilities, managing latency during transitions, and handling conflicts in the actuation process.

4.3.1.1. Registry

This function is to register and to discover DT components based on their feature information, such as metadata, schema, parameters, and descriptions. Key requirements for this system include the comprehensive management of access to DT components, which encompasses things, assets, objects, contents, data, analysis models, and functions. This function supports the seamless registration of new DT components, allowing them to be easily added to the system. Furthermore, effective discovery mechanisms based on detailed feature information are also crucial, enabling users to quickly find and utilize the appropriate DT components for their specific needs.

In the functional architecture, the Registry extends these capabilities by the underlying digital-twin data spaces (Digital Twin Data Storage and Device/Non-Device Data) that support them. Specifically, it provides mechanisms to register and track data sources, maintain links between physical-world data streams and their digital representations, and orchestrate synchronization across heterogeneous environments. Furthermore, the Registry exposes interfaces that allow analytics functions to retrieve consistent contextual data and enables actuation modules to interact with physical objects through well-defined component endpoints.

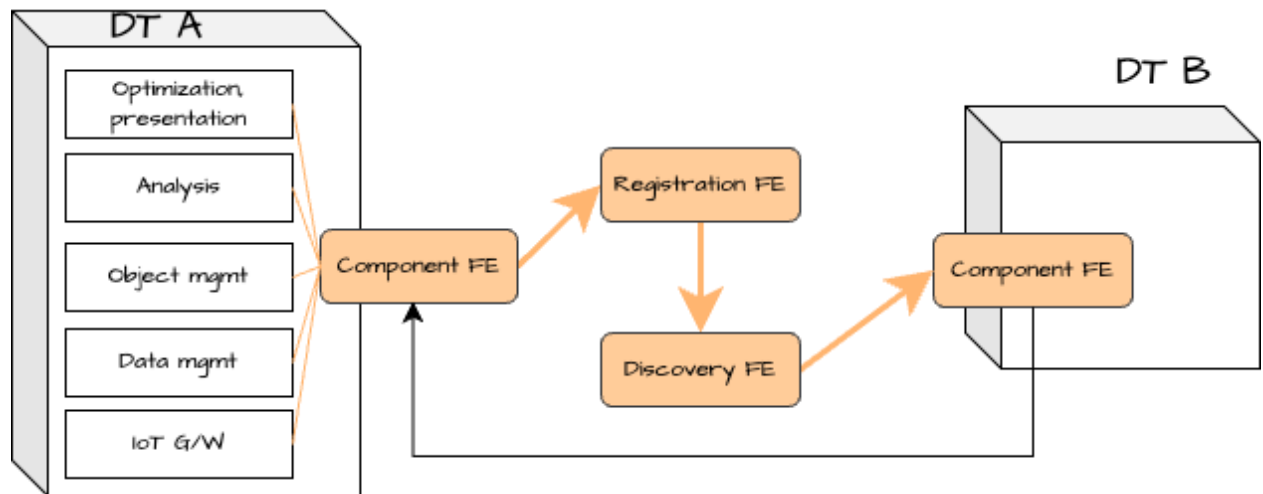


Figure 5: Function elements of the registry function

The figure above shows the Function Elements (FEs) of registry function including Registration FE, Component FE and Discover FE. Component FE is for DT to manage components belonging to the digital twin owner including data objects, analysis objects and simulation objects. The registration function element provides function for developer to register a new component, update a registered component, and remove registered a component of the digital twin. The discovery function element enables other digital twins to locate components registered in the catalog and request access to the functions provided by those components.

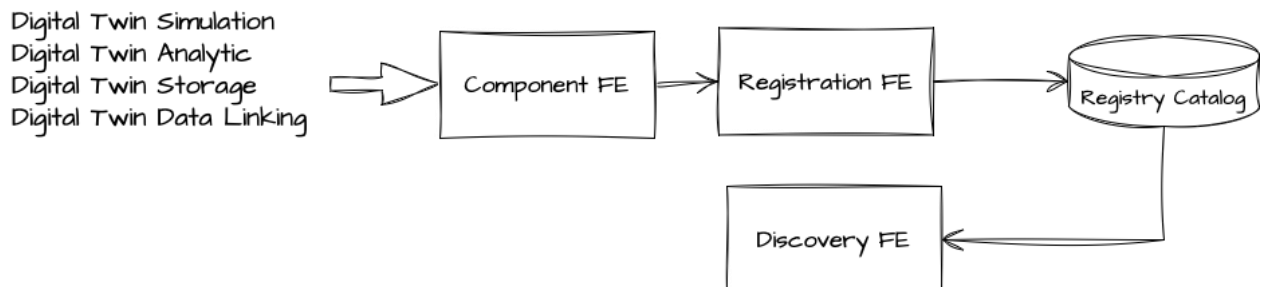


Figure 6: Workflow for the registration of Digital Twin components.

The figure above shows the registration workflow of this function. When registering for a digital twin component, for example an analysis model, Component FE describes the component including model metadata and description. Component FE then requests Registration FE to register the DT component in the registry catalog. After registering, other digital twins and applications can discover and leverage this model.

4.3.1.2. Federation

Configuring and managing a federation of digital twins involves creating a system where individual digital twins contribute to shared virtual models while maintaining the privacy of data generated by physical objects and preserving autonomy across multiple digital twin. The requirements

for such a federation include the configuration and management of federation tasks, ensuring that each digital twin can participate effectively. This involves the distribution, execution, aggregation, and optimization of common models to provide accurate and comprehensive virtual representations. Federation across digital twin includes Federated Learning (FL), Federated Data Mining (FDM), Federated Database (FDB) and Federated Simulation (FSM).

Furthermore, regarding federation across platforms operated by multiple stakeholders, the federation function must provide mechanisms that ensure a trusted collaboration process. This includes capabilities for security management, stakeholder identification, and contract negotiation, enabling secure and reliable interconnection of digital twins across organizational boundaries.

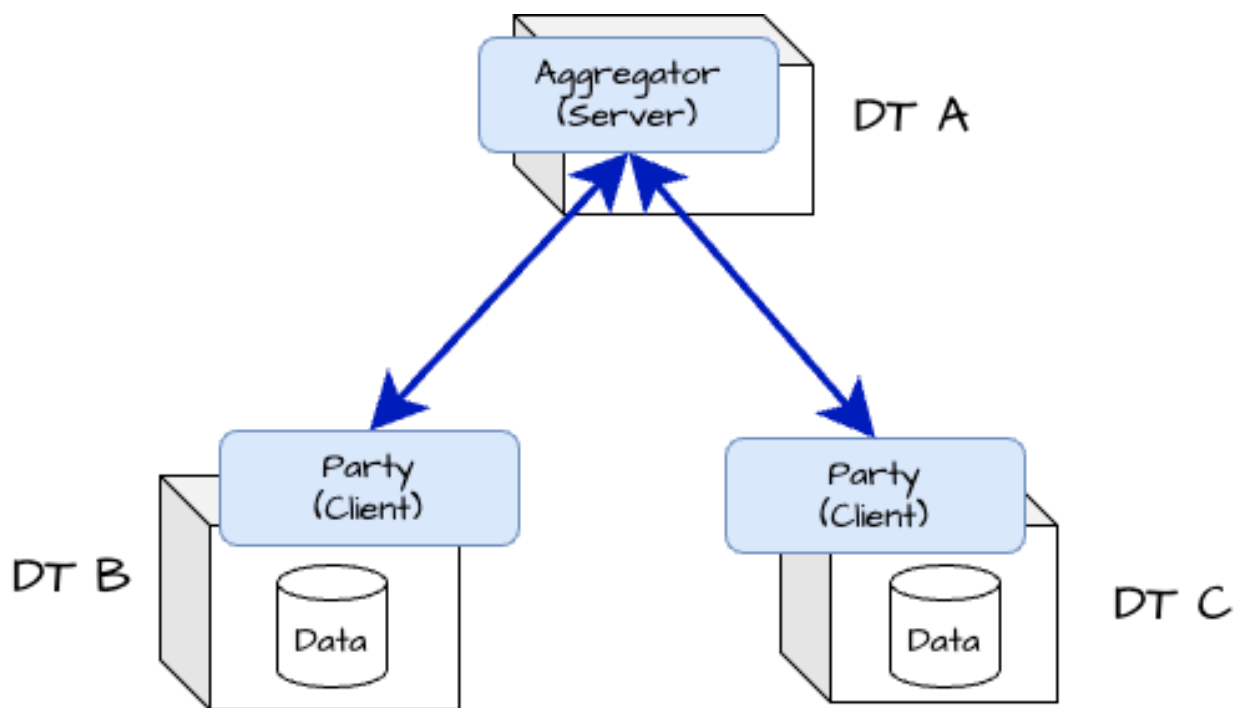


Figure 7: Function elements of the federation function

The figure above shows typical function elements of federation including Aggregator FE and Party FE. Aggregator FE may aggregate models in FL, aggregate insights in FDM, integrate multiple autonomous database in FDB and coordinates multiple simulation objects of digital twin.

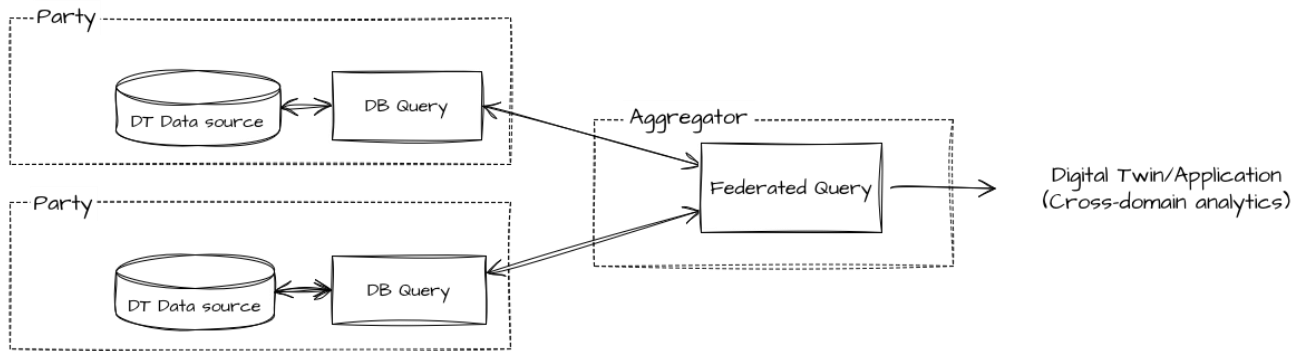


Figure 8: Federated Database (FDB) workflow

The figure above shows workflow of FDB, which is highly applicable in Data Spaces for Digital Twin for integration of multiple autonomous digital twin data source. It includes Digital Twin data source connection with real-time data providers; DB query for dynamically access the relevant data from each source; and Real-time Processing and Aggregation via Federated Query that processes, aggregates and combines the results from different sources to provide a comprehensive output.

4.3.1.3. Translation

Translation in the context of digital twins involves facilitating the syntactic and semantic transformation of data inside a digital twin data space across different domains. Key requirements for effective translation include the conversion of data formats, ontology, and units/scales to ensure consistency and compatibility. Moreover, translation must also support broader automatic data model interoperability, including graph-based models defined by NGSI-LD, OWL, or network-oriented data models with YANG. The translation API should enable dynamic integration of heterogeneous data models in the digital-twin lifecycle.

There are several methods of translation that can be exposed. Schema matching is essential to align different data structures, enabling seamless data exchange. Ontology inference helps to establish common understanding by interpreting and mapping domain-specific terminologies. Machine learning translation further enhances this process by leveraging machine learning model to improve accuracy and efficiency in transforming data between digital twins, thus ensuring smooth interoperability across diverse digital twin data spaces.

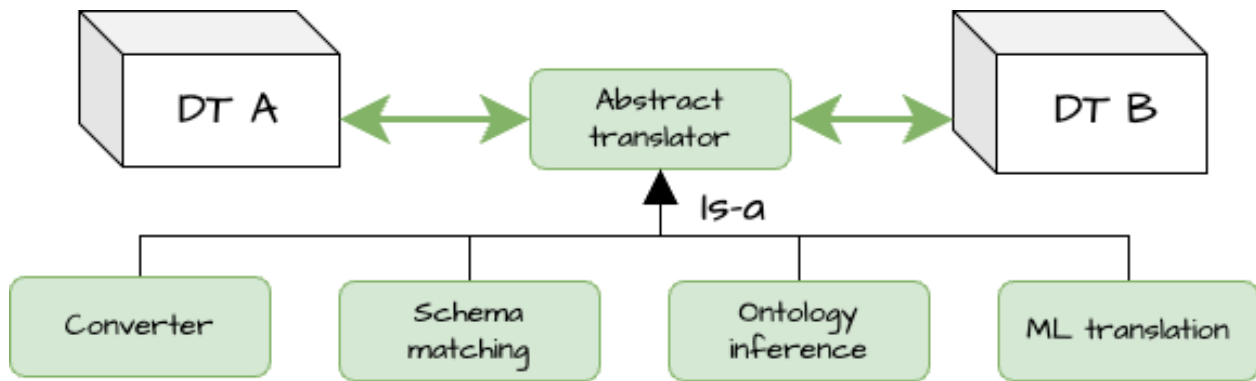


Figure 9: Translation between two digital twins.

Usually, an abstract translator is predefined, as illustrated in the above figure, which can take any function in the form of a heuristic converter, a schema matching module, ontology inference engine, or a machine learning based translation.

Translation can be applied to data exchange between digital twins and devices (device data) as well as between a digital twin and an external system including other digital twin (non-device data). In the functional architecture, the translation function should interact with the synchronization function to ensure reliable data interchange from external systems - both device data and non-device data - into the digital-twin storage.

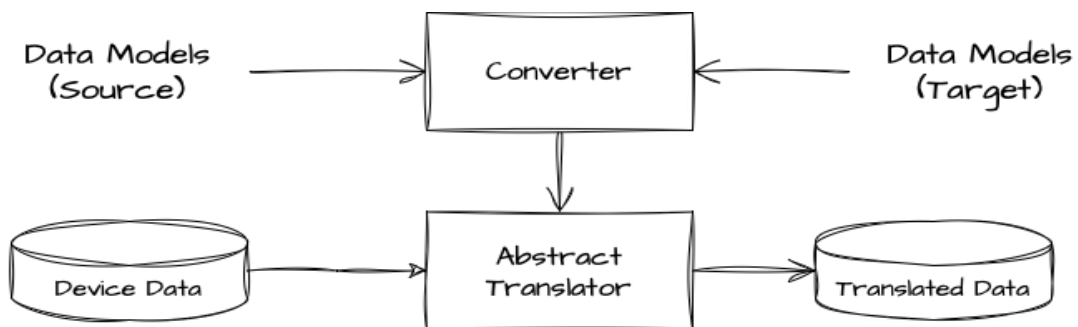


Figure 10: Translation function workflow.

The figure above illustrates a potential workflow for this function. For example, when translating data from an environmental sensor mounted on a vehicle (in-vehicle sensors) into the format used by environmental observation stations, data models define a converter that transforms the source data (formatted for in-vehicle sensors) into the target data format (used by environmental observation station sensors). This converter is embedded within the abstract translator. As data is streamed to the digital twin, it is automatically translated by the converter.

4.3.1.4. Brokering

The brokering function identifies and authenticates digital twins, relays information transmission and reception, and performs data filtering, real-time delivery, and guaranteed delivery is crucial for seamless digital twin operations. The requirements for such a system include the efficient

transmission of data streams and commands, ensuring that information flows smoothly between digital twins. Routing and filtering based on application context are essential to direct the correct information to the appropriate digital twins, enhancing the relevance and accuracy of the communication. Real-time delivery is vital for time-sensitive applications, while guaranteed delivery ensures that critical data is always received. Additionally, the brokering must support internet-scale scalability to handle a large number of digital twins and the vast amounts of data in digital twin data spaces, ensuring robust performance across global networks.

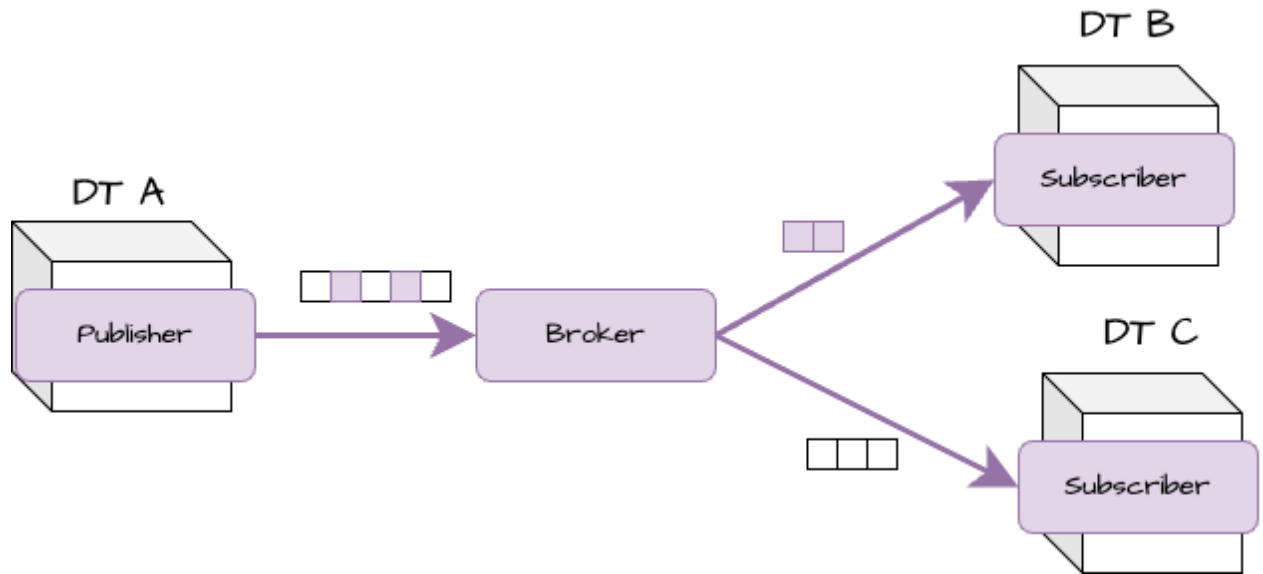


Figure 11: Function elements of the brokering function.

The figure above shows the function elements of brokering including Publisher FE, Broker FE and Subscriber FE. Conventionally, broker is back-end system that coordinate data transmission between digital twins. In the context of digital-twin federation, the broker also supports mechanisms that ensure a trusted collaboration process within the Federation function.

In addition, a Broker can be a context broker (e.g. IDH Context Broker [IOWN-GF-IDH]) which manages subscriptions, filter context information and notify the subscriber (or consumers) when information is updated. This ensures that updated device and non-device data synchronized into the digital-twin data storage can be reliably consumed by the Analytics function, as described in the functional architecture.

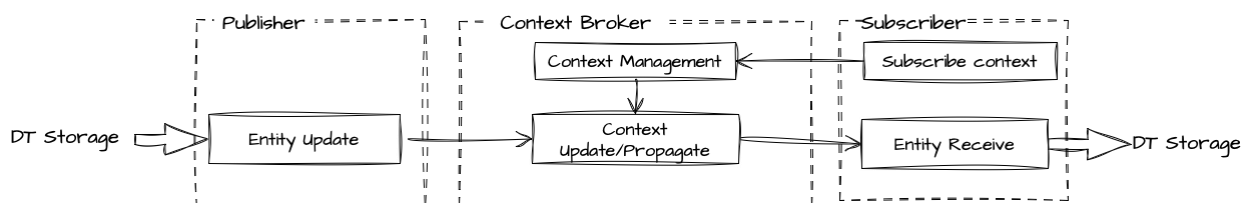


Figure 12: Workflow of the broker function.

The figure above illustrates the workflow of the context broker function. For example, Smart Driving Digital Twins can subscribe to receive emission restriction plans for specific area. When the Emission Simulation Object of a Smart Environment Digital Twin updates or publishes a new emission restriction plan, the context broker updates the relevant context entities (such as the areas with emission restrictions) and notifies subscribed Smart Driving Digital Twins accordingly.

4.3.1.5. Synchronization

Synchronization of multiple interactions with physical entities between digital twins is vital for maintaining consistency and coherence in a digital twin ecosystem. The requirements for this synchronization include the precise identification of physical entities, ensuring that the same physical object referred to by cyber objects in different digital twins is correctly recognized. This involves establishing a reliable method for identifying and linking these entities across various digital twins.

Additionally, the system must coordinate status updates (input/output) of cyber objects in different digital twins that refer to the same physical entity. This includes detecting and resolving conflicts that may arise from updates, such as changes in object locations, status or ids, ensuring that all digital twins maintain a consistent and accurate representation of the underlying physical entity. The synchronization function also interacts with the Translation function to ensure interoperability, including alignment of data models, schemas, and ontologies during data interchange.

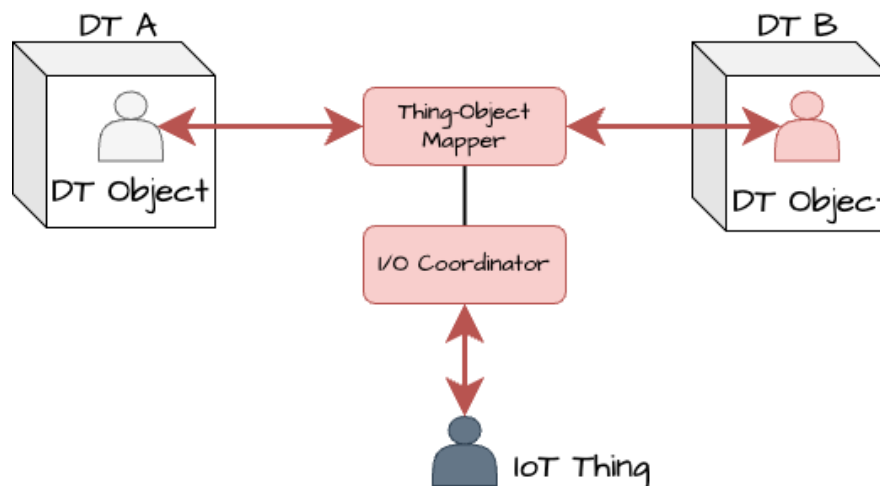


Figure 13: Elements of the Synchronization function

The figure above shows function elements of Synchronization including Thing-Object Mapper and I/O Coordinator. Thing-Object Mapper identifies the same physical entity but refer to different cyber objects in different digital twins. I/O Coordinator will control updating thing description (information of cyber objects) from physical entities, requesting synchronization between digital twins.

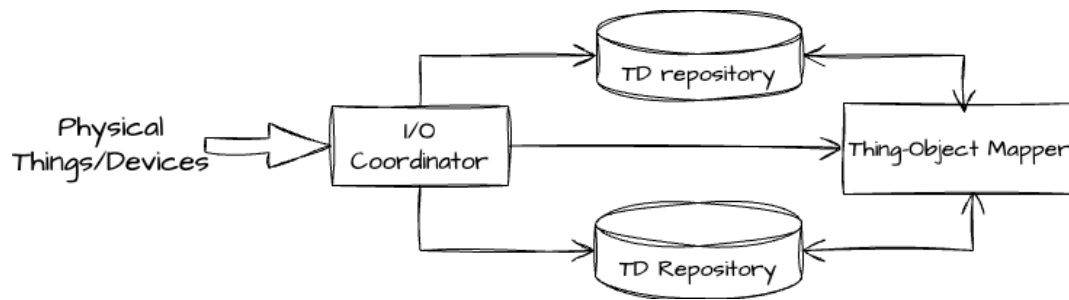


Figure 14: Synchronization function workflow.

The figure above illustrates a possible workflow for this function. The Smart Driving Digital Twin and the Smart Transportation Digital Twin establish synchronization for the physical vehicle's position. When the vehicle (a physical entity) changes its position, the Thing Descriptions in both digital twins are updated through the I/O Coordinator. If there are differences in the Thing Descriptions between the two digital twins, the I/O Coordinator requests the Thing-Object Mapper to synchronize and resolve these discrepancies.

4.3.1.6. Actuation

The *actuation function* is in charge to reflect the status of the digital twin into the real asset. Although there might be multiple digital twin replicas of the same asset in the digital world, there might be one digital twin replica that is synchronized with the real asset: a change in the real world is a change in the digital twin replica and a change in the digital twin replica is mirrored into the real asset. Not every parameter changes of the digital twin can be enforced into the real asset because that depends on the capabilities of actuating such change. The capabilities of actuating a parameter need to be registered into the *Registry function* before an actuation can take place. In the real world, an actuation cannot happen in time null, therefore there will be a time window where there is a mismatch between the digital twin parameter value and the real asset value. The Actuation function is in charge of keeping this transition status within the required latency and to keep the transition status up to date in the digital twin. The transition status is continuously reported and updated into the digital twin representation handled by the brokering function. Thus, eventual mismatch between the digital replica and the real asset is always reported and available through data query or subscription to the data broker. Further the Actuation will be in charge of cope with conflicts of actuation processes.

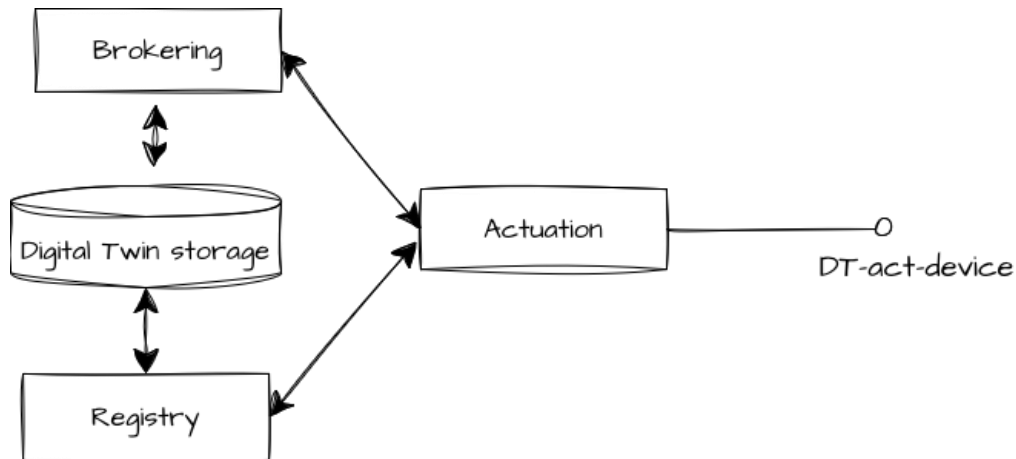


Figure 15: Actuation function overview.

An example workflow is as follow:

1. An actuation capability is registered through the *Registry function*. The actuation capability is attached to one or more parameters of the Digital Twin. The actuation capability reports the interface to interact with the device and other important information such as possible conflicting actions or impossible parameter values (e.g., out of range or conflicting with other parameters).
2. A digital twin parameter linked to an actuation capability changes and the *Brokering function* notifies the *Actuation function*.
3. The *Actuation function* discovers the actuation capability linked to the parameter changes and is instructed on how to actuate the Digital Twin parameter change.
4. In case of conflict with ongoing actuation procedures, the *Actuation function* puts the current actuation in the queue and marks such on-hold status as metadata of the changed parameter in the broker. In case the changed parameter is an unactionable value the *Actuation function* marks such value as faulty through the *Brokering function*.
5. When the actuation process starts, the *Actuation function* calls the device actuation capability as discovered into the *Registry function* and keeps monitoring the actuation (either through the DT-act-device or through related data coming from the *Brokering function*). The actuation status is constantly updated into the digital twin instance and retrievable through data query or subscription.
6. When the actuation procedure terminates (either with a success or failure) the *Actuation function* reports it into the digital twin instance.

4.3.2. Data Analytics

This group of functions facilitates the handling of data analytics functions in complex system with multiple processing nodes (i.e. *Analytics Orchestration function*) and in the data space with multiple stakeholders (i.e. *Cross-Domain Analytics function*). Further, this group also includes functions to improve the Digital Twin analytics (i.e., *Probabilistic Digital Twin function*).

4.3.2.1. Analytics Orchestration

This functional block targets to minimize the effort from digital twin developers and application developers. The Analytics Orchestration function exposes a digital twin data programming model to a digital twin developer that offers a entity-centered specification of the Digital Twin analytics, where a real asset and each of its parts have a uniquely identifiable digital entity. While the user needs only to specify input and output types of the analytics function and how the analytics functions are plumbed in a high-level topology for incremental analytics, the Analytics Orchestration will automatically compute the actual amount of instances and it will handle the lifecycle of the analytics process (from data gathering till process execution).

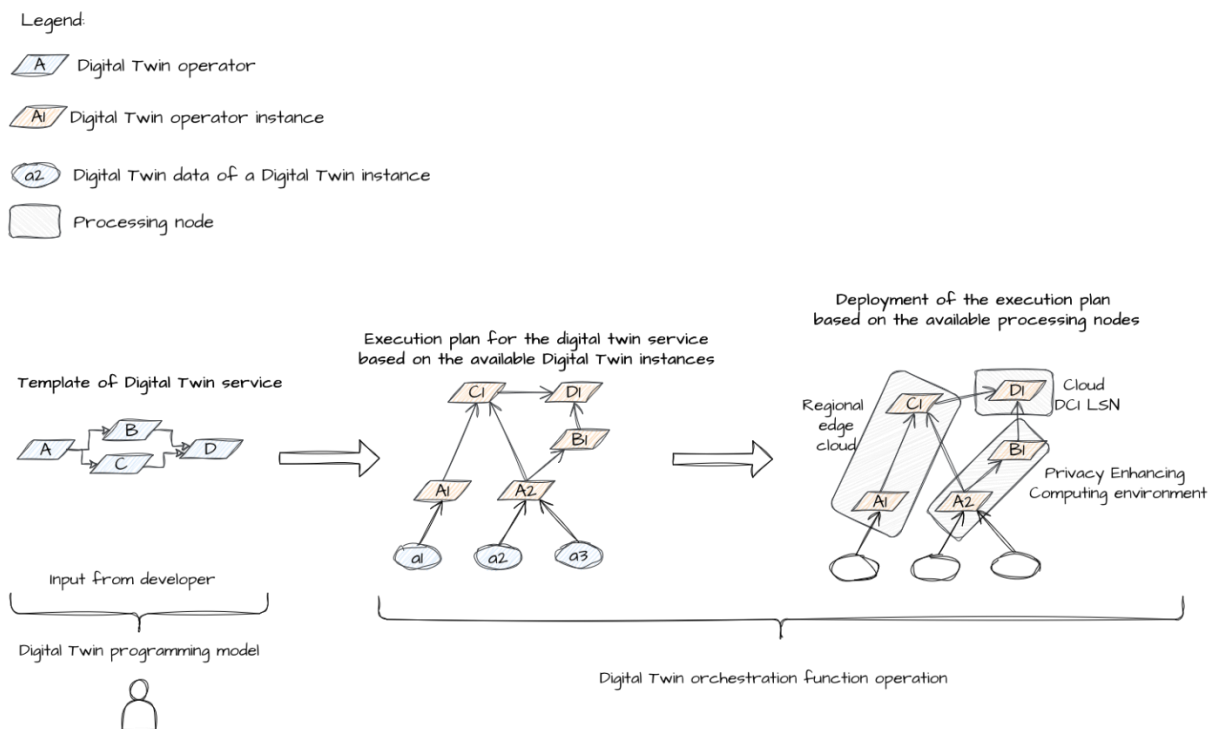


Figure 16: Digital Twin orchestration function operations.

Terminology:

- Digital Twin operator: is a processing function that analyzes data related to a digital twin instance and output new information related to the same digital twin instance. Such operator is registered in the Registry function as element of a Digital Twin.
- Digital Twin class refers to a group of digital twins with common set of properties and behaviours.
 - E.g.: In NGSI-LD [ETSI-GS-CIM-009] it might be modeled as the entity type
- Digital Twin instance is a specific instance of the digital twin class.
 - E.g.: In NGSI-LD [ETSI-GS-CIM-009] it might be modeled as the entity id

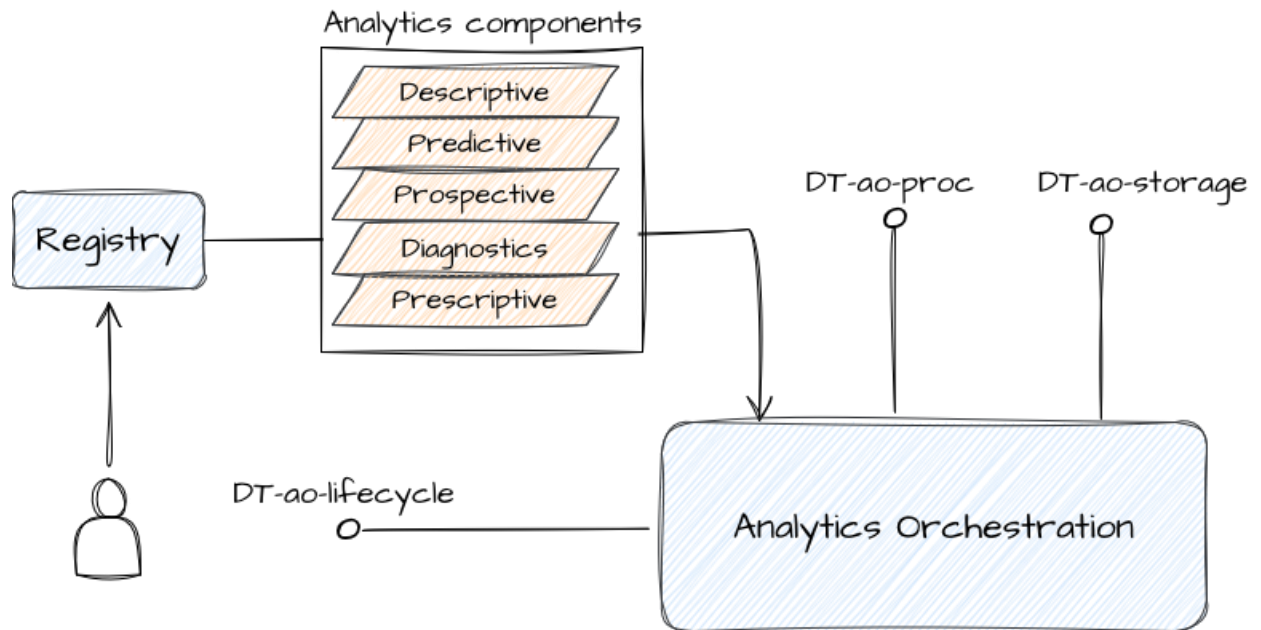


Figure 17: Digital Twin orchestration function interface overview

This functions exposes four interfaces:

- Register (Digital Twin analytics orchestration - user) interface that is used by the Digital Twin developer to associate a digital twin operator to a class of digital twins or a specific digital twin instance. The user can specify different usage context such as a single operator instance per digital twin class or one operator instance per digital twin instance.
- DT-ao-lifecycle (Digital Twin analytics orchestration - lifecycle) interface allows an external user to identify the instantiated processes and directly command them, such as: discover, start, stop, restart, delete. A user can discover the instantiated processes linked to the registered services and directly command with them.
- DT-ao-proc (Digital Twin analytics orchestration - processing) interface for interacting with the processing nodes such as DCI and PET.
- DT-ao-storage (Digital Twin analytics orchestration - storage) interface for interacting with the Digital Twin storage

The Digital Twin analytics orchestration function uses the information given through the DT-ao-user interface to instantiate the adequate number of digital twin operators according to the available instances of Digital Twin. The available instances of Digital Twins are retrieved through the DT-ao-storage interface. The actual instantiation of the operators in a processing node is done through the DT-ao-proc interface. For example, Digital Twin analytics orchestration uses this interface to request a new logical service node (LSN) to DCI [IOWN-GF-DCI], to discover privacy-enhanced computing nodes, and to command the instantiation of a service into a processing node (e.g., LSN or a PET). After the request, the DCI cluster returns a bare-metal DCI LSN which will be configured to host the Digital Twin analytics service by the Digital Twin orchestration function.

A possible workflow can be similar to the following:

1. A digital twin developer implements a Digital Twin operator and registers it through the Register function
2. A digital twin developer associates the Digital Twin operator to a Digital Twin class specifying usage context (e.g., geolocation of digital twin instances, specific digital twin instance of such class, digital twin instances with a specific attribute set)
3. The Digital Twin analytics orchestration discovers all the instances available of the specified classes through the DT-ao-storage.
4. The Digital Twin analytics orchestration generates a plan on the all needed instances of the digital twin operators depending on the discovered available instances (point 3.) and specified usage context (point 2.)
5. The Digital Twin analytics orchestration uses the DT-oa-proc interface to discover suitable processing nodes and to require the instantiation of new processing nodes if needed
6. The Digital Twin analytics orchestration makes an execution plan by associating an operator instance to a processing nodes. The execution plan strategy might include different factors such as privacy/security, performance, and energy.
7. The Digital Twin analytics orchestration commands the instantiation of operators according to the execution plan via the DT-oa-proc interface.
8. The Digital Twin analytics orchestration allows the discovery of instantiated processes associated with a registered service.
9. A digital twin developer can directly commands the instantiated processes such as stop, start, restart, or cancel.

4.3.2.2. Analytics monitoring

This functional block targets to provide observability, accountability, and results accessibility for Digital Twin simulation workloads executed across heterogeneous processing environments (e.g., DCI LSNs, PET nodes, cloud/edge runtimes). The **Analytics monitoring function (DT-am)** offers a uniform, technology-agnostic programming model to monitor simulation executions, retrieve runtime statistics, and export results in interoperable formats. The function abstracts the underlying execution platform and exposes consistent semantics for simulation lifecycle visibility (queued/running/completed/failed), progress tracking, resource usage, and outputs availability. In doing so, DT-m-proc reduces the effort for Digital Twin developers and application developers to integrate simulation observability and result retrieval into smart solutions, while enabling governance-friendly reporting, auditing, and downstream reuse of simulation outcomes.

Terminology:

- Digital Twin operator (see Analytics Orchestration function): an executable workload that runs an analytics operator such as a simulation model for one Digital Twin instance, a

Digital Twin class, or a scoped subset of instances, producing results and/or derived statistics.

- Analytics instance: a uniquely identified execution of an analytics process, parameterized by configuration (inputs, model version, time horizon, scenario).
- Analytics status: the runtime state of an analytics instance (e.g., *registered*, *queued*, *provisioning*, *running*, *completed*, *failed*, *cancelled*), potentially enriched with progress indicators and timestamps.
- Analytics statistics: runtime and outcome metrics associated with a analytics instance (e.g., duration, iterations, convergence, CPU/memory usage, I/O, energy estimates, produced output size, quality indicators).
- Export artifact: a packaged representation of results and/or statistics for external consumption (e.g., JSON document, RDF graph), optionally including metadata (provenance, model version, parameters, timestamps).

DT-am exposes the following interfaces:

1. DT-am-status (Monitoring API)
Provides monitoring access to simulation process status, progress, and lifecycle events. It supports both pull-based queries and push-based subscriptions to enable near-real-time observability.
2. DT-am-stats (Statistics API)
Provides access to runtime and outcome statistics for simulation runs, supporting aggregation over time windows and filtering by Digital Twin class/instance, scenario, model version, or execution node.
3. DT-am-export-json (JSON Export API)
Provides export capabilities for simulation results and/or statistics in JSON-based formats suitable for application integration, dashboards, archival, and data-space sharing.
4. DT-am-export-rdf (RDF Export API)
Provides export capabilities for simulation results and/or statistics in RDF, enabling semantic interoperability, linking to Digital Twin entities, provenance tracking, and cross-domain knowledge integration.

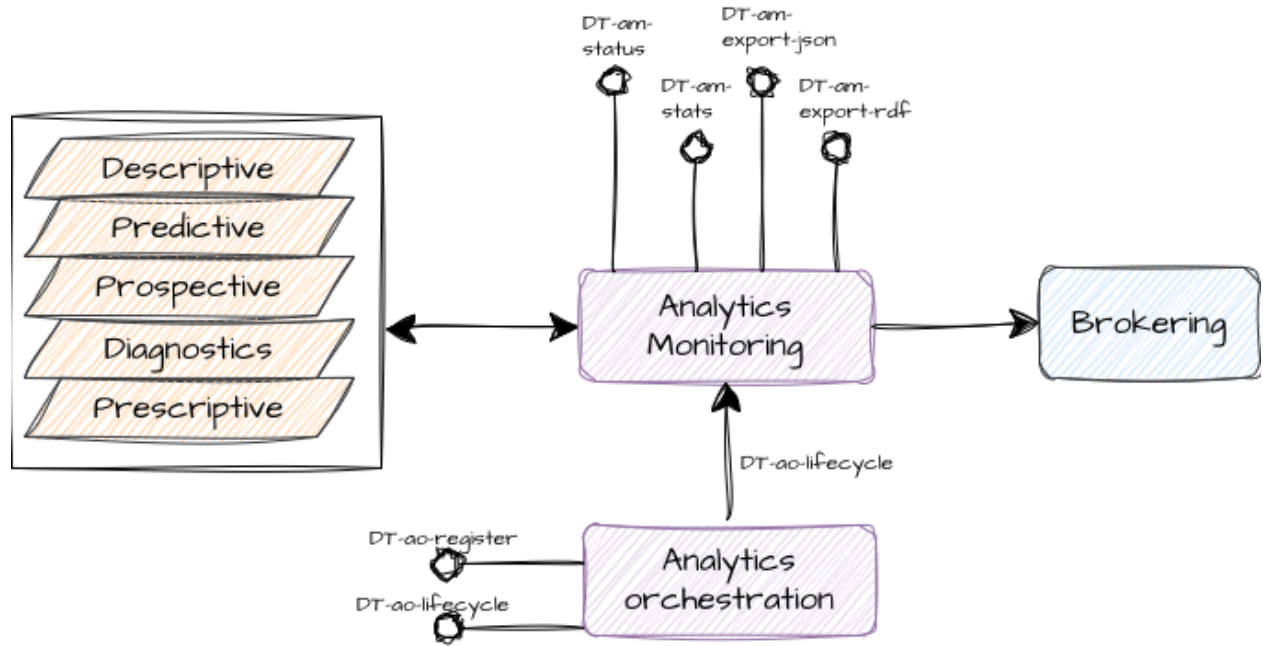


Figure 18: Analytics monitoring function interface overview.

DT-am integrates with simulation execution environments via the processing layer (e.g., orchestration/compute services). Once a simulation run is started by the relevant orchestration or execution function, DT-am maintains an updated view of its lifecycle by either:

- receiving execution events from the processing platform (push), and/or
- polling execution runtimes for status updates (pull), and then exposing normalized monitoring and export interfaces to external users and applications.

The simulation status is continuously synchronized such that any mismatch between the reported execution state and the actual runtime state is detected and surfaced through status queries and/or subscriptions.

An example workflow is as follow:

1. A Digital Twin developer triggers a simulation run (via the DT-ao-register or DT-ao-lifecycle).
2. DT-am discovers the run through DT-ao-lifecycle and begins tracking it (state: *observed/queued*).
3. An application subscribes to status updates for the instance or a filtered set of instances (e.g., all instances for a Digital Twin class and scenario).
4. DT-am reports state transitions (*provisioning* → *running* → *completed/failed*) and progress updates.
5. A user queries statistics for the completed run (duration, resource usage, outcome KPIs).

6. The user exports results and/or statistics as JSON for downstream ingestion into dashboards or data-space sharing.
7. Optionally, the user exports RDF to publish semantically interoperable results linked to Digital Twin entities and provenance metadata.

4.3.2.3. Cross-domain analytics

Cross-domain analytics functional block targets the digital twin interoperability requirement *DTF-Req-5* [IOWN-GF-DTF-AR] with means of automatic collaborative data analytics across federated domains of stakeholders. The functional block enables an automated data analytics marketplace based on semantic data of multiple digital twins. In this context, the semantic data consists of raw data and semantic annotations, which can be represented as a graph, 3D models, or any other applicable format. For the sake of explaining the enabler, one can consider vendors are the stakeholders that use the IOWN digital twin framework with the enabler on cross-twin semantic analytics, where different stakeholders in cross-domains have different digital twins. The stakeholders collaborate through the IOWN DSDT for improved application services.

Terminology:

- Domain: partition of the DSDT system owned by a stakeholder of the data space.

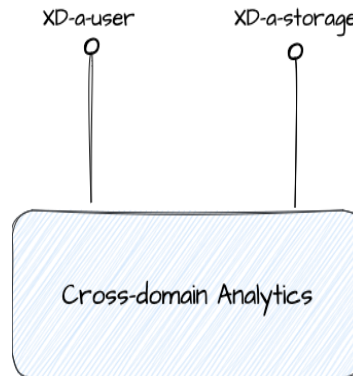


Figure 19: Cross-domain analytics interface overview

For this functional block users can be served automatically where they can enter manual or automated queries for their needed analytics tasks. A stakeholder can be a provider of data, a consumer of data, or both of them. The involved interfaces are:

- XD-a-user (Cross-domain analytics user) interface is used by a stakeholder (or its application) to request the needed data. The data might currently available as it is or it needs to go through a processing function (e.g., data preparation, data aggregation or, even, ML based). To avoid ambiguities, the data requested follows a semantic approach (e.g., through an ontology). The requested data is accompanied by a description of it such as

expected quality, expected Digital Twin instances, expected delays, maximum costs (e.g., in terms of energy or dataset value).

- XD-a-storage (Cross-domain analytics storage) interface is used to discover the Digital Twin data into the federation of multiple domains. The data is supposed to be semantically annotated through the translation functional block.

Previous to any call to cross-domain analytics, one or more provider register through the *Registry function* the availability of a Digital Twin component that is either data or a feature for the Digital Twin (e.g., an analytics process that infer a value of the Digital Twin). The provider together with the availability of the data or features register also terms of usage of such component such as cost, energy expenditure, quality of the data, privacy, license. At this point, once a consumer request for such data or features through the XD-a-user interface, the *cross-domain analytics* function browse among all the matching Digital Twin components, select the matching ones and form the best fitting pipeline of template for the consumer. In case of more than one component is matching for the same purpose, the *Cross-Domain analytics function* will either choose on the given parameters (such as based on costs or quality) or the *Cross-Domain analytics* function might combine them. The policy for multiple matching components can vary between implementations. The best fitting pipeline can be generated through the application of an optimization problem maximizing or minimizing an object function (minimize energy, minimize costs, maximize quality, etc.). The pipeline will involve the usage of data and feature from different stakeholders in the federation. Once the best pipeline (that can be composed also by a single feature) is decided, the *cross-domain analytics* submit it to the *Digital Twin analytics orchestration* by using the *Registry function*. The *Digital Twin analytics orchestration* will then generate an execution plan and execute it.

A possible workflow can be similar to the following:

- A provider register a Digital Twin component that might include:
 - Output (Predictions or analytics results)
 - Historical inputs required for training (located in the IDH Data Hub)
 - Live inputs required for executing
 - Model quality
 - Terms of usage/access
 - Other characteristics
- When a consumer requests a Digital Twin component the *cross-domain analytics* determines the currently accessible portion of Digital Twin and which providers offer it. The best fitting Digital Twin component or chain of Digital Twin components is chosen and send to the *analytics orchestration* through the *Registry function*.
- The *analytics orchestration* execute the necessary components and retrieve the data to the consumer.

4.3.2.4. Probabilistic Digital Twin

The scope of Probabilistic Digital Twin is to manage real-world Digital Twin data and Network Digital Twin data in a probabilistic way because safety and trustability are critical for many Digital Twin applications. Probabilistic Digital Twin offers a way in which risk management can be better handled through probabilistic representation of the real world, and probabilistic prediction and probabilistic control based on the probabilistic representation.

The Probabilistic Digital Twin function, as one of the Digital Twin analytics enablers, should have an interface to access the digital twin data stored in the Digital Twin storage. The Probabilistic Digital Twin function expects that the storage allows for managing probabilistic nature of the Digital Twin data. The following figure shows one example of such probabilistic data that represent the spatio-temporal structure of a space. An object, which is one data component in the space, may be a physical thing, such as a human, robot, and car, in the space (orange dots) or a status of a lattice point in the 4D space (green dots). The identity or location of the objects, which are inferred by any AI/ML algorithms from raw observation data, should include uncertainty no matter how the AI/ML algorithms evolve, and thus such information should be handled as a probabilistic distribution, rather than "best estimates".

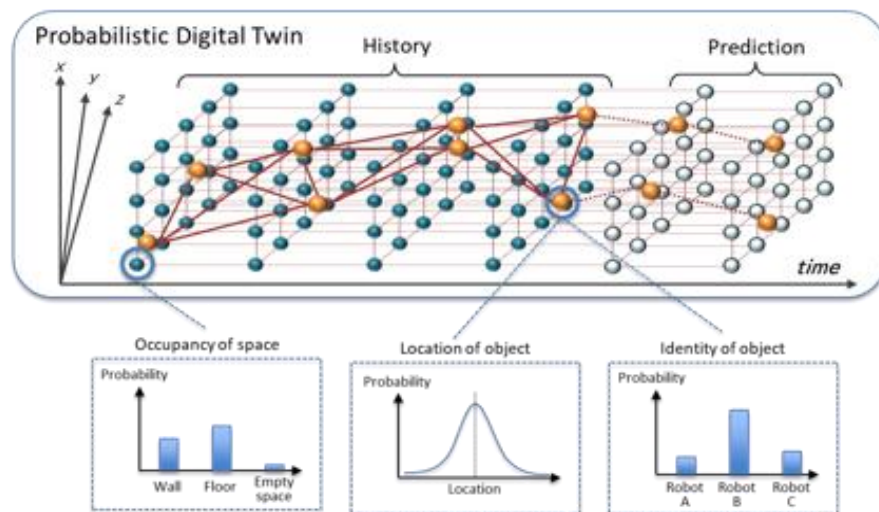


Figure 20: Example of Probabilistic Digital Twin Data Structure

The following are three patterns of functional block operations as shown in the following figure. The operations are similar to the operations of Translation, Synchronization, and Linking functions, and thus it might be possible to think that the Probabilistic Digital Twin function is one particular combinational realization of those functions.

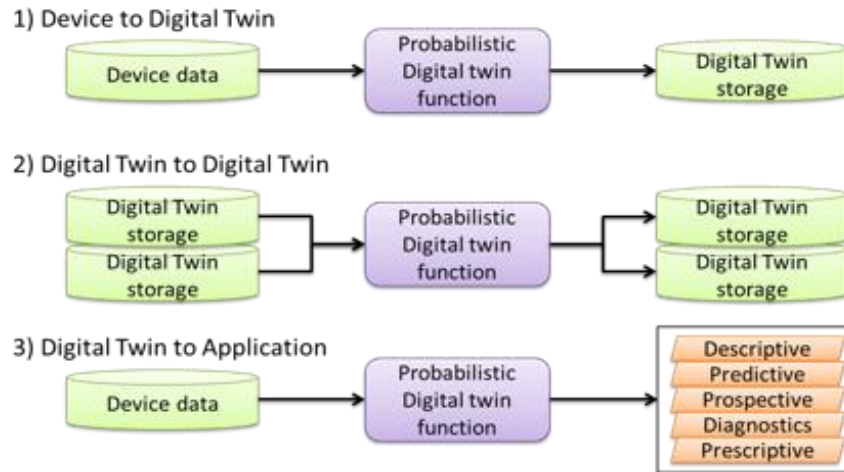


Figure 21: Functional block patterns of the Probabilistic Digital Twin function

1. Device to Digital Twin

Environmental observation data is analyzed and its results are stored in Digital Twin storage as statistical information. For example, the function receives video data from a camera, or point cloud data from a LiDAR device, and recognizes objects from using the data and estimates their class labels and locations. In this case, the class label should not be a best-estimated class label, but rather a probability distribution of class labels. Also, the location should not point to a specific location in a space, but a probability distribution in a 3D space.

2. Digital Twin to Digital Twin

One or more Probabilistic Digital Twin data might be converted to one or more Probabilistic Digital Twin data through spatio-temporal inferences. One such spatial inference may be multimodal recognition, in which one Digital Twin created from camera data and other Digital Twin created from LiDAR data are inferred jointly together and new Digital Twin data are produced as a result of multimodal recognition. Another example may be temporal inference, in which series of Digital Twin data are analyzed and then prediction of person movement, future location of cars, and collisions are generated as specific Digital Twins.

3. Digital Twin to Application

Finally, the Probabilistic Digital Twin data are provided to any Digital Twin applications as a Descriptive, Predictive, Prospective, Diagnostics or Prescriptive Digital Twin. The function may send the Digital Twin data as a form of probability distribution. The function may perform some statistical operations as applications require and send a joint probability or marginal probability of a specific event. If the application is not capable of statistical operations, the function may perform maximum likelihood estimation and send it to the application as the best estimate result.

4.3.3. 4.2.3 Digital Twin manipulation

The Digital Twin manipulation function block describes views for most common interactions between the user and the digital twin entity, consists of:

- The **Simulation** involves creating a virtual replica of the physical system or object. It includes modeling the behavior, characteristics, and interactions of the system using mathematical algorithms and equations. The simulation component allows for testing and analyzing different scenarios and predicting the system's behaviour under various conditions.
- The **Visualization** focuses on presenting the simulated data and information in a visual format. It includes creating graphical representations, such as 2D or 3D models, charts, graphs, or animations, to help users understand and interpret current status of the system, the simulation results, etc. Visualization enhances the user's ability to observe and analyze complex data, patterns, and trends, making it easier to make informed decisions and take appropriate actions based on the simulation outcomes.

4.3.3.1. Simulation

- Receiving parameter input, data processing operator, and performing simulation computing. Data converting functionality is offered by “Digital Twin data handling-*Translation*” (Please refer to 4.2.1). Computing management functionality is offered by “Digital Twin *analytics orchestrator*” (Please refer to 4.2.2)
- Workflow
 - Receiving data and user inputs
 - Conversion of data formats and unit/scale to adapt them to the simulation tools or algorithm that user is using (performed by Digital Twin data handling-*Translation*) .
 - Reserve space and computing environment (performed by Digital Twin *analytics orchestrator*)
 - Execute algorithm or third-party tools for data analysis (performed by Digital Twin *analytics orchestrator* and commanded by Digital Twin *analytics lifecycle*)
 - Continuously monitoring the running analytics (performed by Digital Twin *analytics monitoring*)

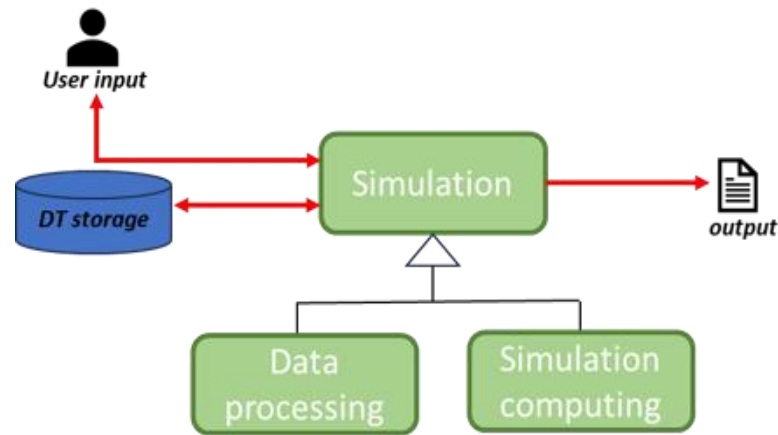


Figure 22: Simulation function overview

4.3.3.2. Visualization

- Receive data and convert the data format according to visualization application requirement. Data converting functionality is offered by “Digital Twin data handling-*Translation*”. Please refer to 4.2.1. Data accessing functionality is offered by “IDH”.
- Requirements
 - Receiving data and simulation result
 - Conversion of data formats and unit/scale depends on the visualization tools that user is using (performed by Digital Twin data handling-*Translation*).



Figure 23 Visualization function overview.

The *Digital Twin manipulation* function exposes three interfaces:

- DT-m-user (Digital Twin manipulation - user) interface which provides the Digital Twin developer to interact with digital twin. Developer can upload the third-party tool or algorithm to develop applications for their field. Also, developer can input parameters to adjust the different simulation scenario and control the digital twin entities. The main functionality of this interface may be offered by the API of the “Digital Twin data handling-*Registry*”. Please refer to 4.2.1
- DT-m-proc (Digital Twin manipulation - processing) interface to control the processing nodes such as DCI and PET. The main functionality of this interface is offered by the “Digital Twin *analytics orchestrator*”. Please refer to 4.2.2

- DT-m-storage (Digital Twin manipulation - storage) interface to access the real time and historical data of digital twin entities, in addition the static data like 3D models. This interface can be offered by IDH such as the “Digital Twin storage” in the overview figure in section 4.1

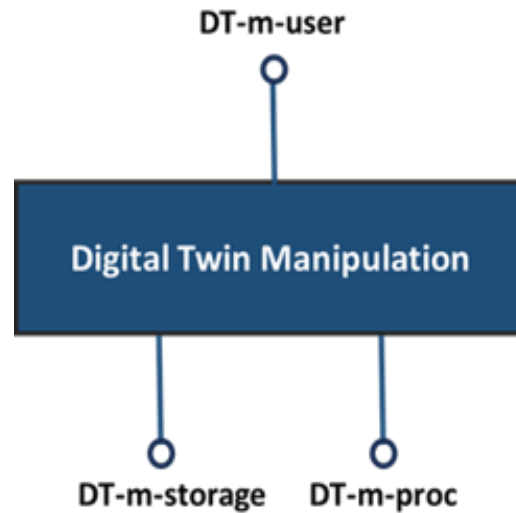


Figure 24: Digital Twin manipulation function interface overview.

A possible workflow can be similar to the following:

1. A digital twin developer uploads the simulation and visualization components in the *Registry function*.
2. A digital twin user interact with the DT-m-user interface (that can be offered by the Visualization component or as a separate API) to input parameters and adjust simulation scenarios.
3. According to the programming, the simulation component obtains the required data through DT-m-storage interface for data processing and simulation computing.
4. Simulation component are executed by the *Digital Twin analytics orchestrator function* to perform data processing and simulation computing. Simulation results are stored through the DT-m-storage.
5. Visualization component obtains simulation result and the required data through DT-m-storage interface. The visualization component presents the data to the Digital Twin user.

4.3.4. Data Space security

IOWN security is essentially a combination of data protection in motion, data protection in use, and data protection at rest. And where and at what cost each protection is applied depends on the requirements of the application on the data space.

4.3.4.1. Data Access control

Isolation mechanisms for data in all states—data in motion, data in use, and data at rest—should be provided in order to protect data from both external attacks and internal threats, including data center operators and other tenants sharing the same data center.

Regarding the protection of data in use and data at rest, it is desirable to consider adopting various isolation technologies mentioned in the Functional Architecture for Protection of Data in Use: IOWN Privacy Enhancing Technologies [IOWN-GF-PET] because even if it is a case of a single stakeholder, data needs to be protected from DC operators and external attacks in some use cases, depending on the analyzed risks.

In each of the multi-stakeholder patterns, data protection needs to be guaranteed based on a data policy agreed upon among the stakeholders in addition to the single stakeholder pattern, since the data is provided to other stakeholders.

- DS-isolation configuration interface configures settings to protect data in motion, data in use, and data at rest. Examples include enabling the Trusted Execution Environment (TEE), configuring encrypted communication, and setting up storage encryption.
- DS-policy management interface to configure access control policies that define which users can access which data and for what purposes.
- DS-authentication interface to provide the security process of verifying the identity of a user or application attempting to access the
- DS-PDP (Policy decision point) authorize interface provides a unified runtime authorization function for control-plane operations, including data subscription and task execution on the requested data. It verifies whether an authenticated subject or application is permitted to perform a requested action on a given resource at the time of execution. By enforcing authorization checks immediately before subscription establishment or task instantiation, the interface ensures that only authorized operations are allowed within the platform, including those initiated by internal orchestration components.

4.3.4.2. Data Usage Control Orchestration

Data Usage Control Orchestration function handles the high-level service requests issued by data consumers, and then translates them into low-level orchestration actions, which can establish a cross-domain data processing pipeline that is compliant with the data usage control policies of data providers.

- DS-submit service intent interface to accept service requests from the application that uses the data.

4.3.4.3. Attestation, tracing/auditing

Each node on which the aforementioned functions are deployed should support capabilities for external verification of its execution environment, as well as capabilities for tracking and auditing processing activities. This capability enables communication nodes and users to verify that the software implementing each function, including security-related functions, is executing in an un-tampered state.

It is recommended that this function exposes the following interfaces:

- DS-attestation interface to verify the configuration of each node or an entire data space. This interface can be used to verify the hardware/devices used, OS and boot-time register information, on-board software, and their configurations (kernel modules, network settings, software settings, etc.) on each node or an entire data space.
- DS-lineage interface to provide traceability functions such as logging and history management of data on each node or a data space.

4.3.4.4. Monitoring for Early Detection against Unknown Attacks

Damage caused by unknown attacks must be anticipated even when the various management and protection measures described above are implemented. For critical data, early detection, response, and recovery capabilities must be prepared. Unknown attacks tend to be detected late precisely because they are unknown, necessitating the preparation of monitoring for early detection.

It is recommended that this function exposes the following interfaces:

- Integrity-Critical Data Monitoring Interface: Compares data on each node or data space against expected values and returns results.
- Confidentiality-Critical Data Access Behavior Monitoring Interface: Detects unusual behavior from data logs or network records on each node or data space and provides audit triggers.

5. Security Considerations

5.1.1.1. Data sovereignty

In each of the multi-stakeholder patterns, data protection needs to be guaranteed based on a data policy agreed upon among the stakeholders in addition to the single stakeholder pattern, since the data is provided to other stakeholders. A secure data space based on the PETs Functional Architecture should be established for data providers and application providers to maintain their respective data sovereignty.

5.1.1.2. Post-quantum security

Security functions used by the above-mentioned functions should be capable of supporting quantum-resistant cryptographic mechanisms. In particular, public-key cryptographic functions, including key establishment and digital signatures, should support quantum-resistant algorithms.

Furthermore, for use cases involving data with particularly high potential impact in the event of leakage, the adoption of the Multi-Factor Security proposed by the IOWN Global Forum is desirable. For the protection of data in motion, especially when considering countermeasures against unknown threats in the age of quantum computers, it is effective to adopt post-quantum cryptographic communication technologies and to provide cryptographic communication based on the Functional Architecture for Protection of Data in Motion: Multi-Factor Security Key Exchange and Management [IOWN-GF-MFS].

5.1.1.3. Resilience (Response to Unknown Attacks)

Critical data requires various protections and secure design approaches. Even with these measures, data that could cause significant business disruption if compromised must be maintained in a resilient state. Prioritize critical data based on the magnitude of business impact if compromised. For high-priority critical data, initial damage mitigation and rapid recovery procedures should be prepared to maintain a resilient state.

6. Interaction with IOWN infrastructure

6.1. Interaction with IDH

For each of the four data patterns described below requires data storage and management functionalities that can be provided by using IOWN Data Hub, or IDH in short.

1. For a single stakeholder using a centralized server, essential functionalities include data preprocessing (such as format conversion and cleaning), data aggregation, effective data storage management, and basic security measures to control access.
2. In a single stakeholder scenario with multiple servers or distributed storage, distributed data management comes into play. In this approach, key functionalities required are data integrity across servers along with scalability and low latency data access.
3. For multiple stakeholders using a single data source that is managed by a stakeholder, this setup requires functionalities like cross-organizational access control, data lineage and provenance tracking to monitor data usage.
4. When multiple stakeholders share each other's data sources, federated data management allows stakeholders to retain control over their data, enhancing privacy and security. Necessary functionalities include data federation and integration, robust security measures including protection of data in motion and data in usage, data lineage tracking, and adherence to interoperability standards.

In each pattern, IDH provides basic data storage and management functionalities, such as:

- **Secure Data Storage**
IDH provides a secure data storage environment. For instance, data is encrypted with a private key controlled by the IDH system owner. This function is used across all four patterns as a common function.
- **Logical Integration of Distributed Data Storage**
IDH logically integrates data stored in the datastores implemented as some IDH Service Class. This means that, up to the extent permitted by security, it is possible to visualize where and what data is stored and provide seamless data accesses regardless of which IDH stores the data. This function is used for patterns 2, 3, and 4.
- **Data Movement and Placement**
IDH streamlines data movement and placement in a distributed environment. This function can be used both within a single IDH system and in a complex system where multiple IDH systems work together. This function is used for patterns 2, 3, and 4.
- **Data Access Controls**
IDH controls data access control within an organization. Data access is permitted based

on the privileges given to the data accessor, data accessor attributes, and attributes of the requested data. This function is used across all four patterns as a common function.

- **Federated Data Access Controls**

On top of IDH data access controls described above, IDH also governs federated data accesses. That means that IDH coordinates with external organizations' identity management and extends data access controls for cross-organizational data access. This function is used for patterns 3 and 4.

- **Secure Data Provision for IOWN PET environment**

In multi-stakeholder scenarios, there are cases where the data itself cannot be shared. However, it is conceivable to share data processing results among multiple parties, ensuring that all sensitive information is removed. Such scenarios would be very useful for effectively realizing the value of data. To support such scenarios, IDH provides a function to securely provision data to the IOWN PET system, or IDH functions as a component of the IOWN PET system. This function is used for patterns 3 and 4.

- **In-place Processing and Analytics**

IDH not only stores data, but also provides the ability to process and analyze the stored data on the fly. Such features can help improve system performance and reduce costs by reducing the need to move data. This function is used across all four patterns as a common function.

- **Data Access, Movement, and Usage Logging**

IDH logs all data access, movement, processing, and analytics activities. Such logs can not only help find out the cause of a data security issue but also help manage data lineage and provenance. This function is used across all four patterns as a common function.

6.2. Interaction with DCI systems

While the IOWN Data Hub handles the distributed data storage across multiple processing nodes, the analytics services of the Digital Twin are managed by the *Digital Twin analytics orchestrator* and require processing nodes to be executed. The Digital Twin analytics orchestrator uses the *DT-ao-proc* interface to request the instantiation of processing nodes or the reservation of existing ones.

When an analytics service is due to be executed, the Digital Twin analytics orchestrator function looks for a processing node to host it. The function will request either a processing node already available or to instantiate a new node depending on the requirements of the Digital Twin applications, the characteristics of the service, and the agreements of stakeholders in the data space.

The Digital Twin analytics orchestrator function would make a request through the DCIaaS interface of the DCI Service Exposure Function for a bare-metal logical service node (DCI LSN) that will host one or more services. The Digital Twin orchestrator function might run some optimiza-

tion algorithm to identify the best solution to host multiple (might be thousands) analytics services in the same DCI LSN. The choice of the DCI cluster depends on the geographic location, the owner of the DCI cluster (in order to comply with data usage control constraints), or a combination of those. Further, the Digital Twin analytics orchestrator function will request the DCI LSN assessing the QoS requirements of the Digital Twin applications. In future development of the DCIaaS (DCI as a Service) interface, the DSDT function would request a DCI LSN specifying the QoS and the DCIaaS will identify the required processing node specifications.

To cope with the data sovereignty of the data space, DSDT might enforce the execution of certain analytics services into processing nodes that support Privacy-Enhancing-Technologies (PETs). Therefore, when requesting a processing node from the DCIaaS interface, the *Digital Twin analytics orchestrator* would request DCI LSNs that include PETs, e.g., processors that support TEEs.

In order to allow the interaction with data centers managed by the IOWN DCI, a DSDT system might implement two optional functions that would address the following requirements:

- A DCI cluster do not discloses the available resources in the cluster such as computation power
- The server requirements needed to run the Digital Twin computation and storage for the particular application and constraints (such as latency).

6.2.1.1. (Optional function) Privacy-Preserving Node Discovery & Allocation Broker

This component enables the DSDT to obtain suitable compute capacity without requiring data centers to disclose node inventories. It provides a request/offer interaction model where the DSDT submits a node request (constraints + preferences), and each participating compute space (e.g., a DCI cluster operator) replies with an accept / reject decision, optionally including a minimal “offer token” (e.g., price class, approximate start time, and capability class) that does not reveal topology or asset details.

At a high level, the component supports:

- Bid-style allocation: the DSDT issues a “bid” (requirements, duration, QoS target, privacy/security constraints), and compute spaces respond with yes/no (and optional coarse-grained offer metadata).
- Iterative negotiation: on rejection, the DSDT can automatically relax or modify requirements (e.g., different accelerator, region, security tier) or redirect the request to other clusters.
- Policy-based filtering: ensure requests comply with governance constraints (data locality, trust level, PET requirement) before submission.

- Opaque provisioning handles: on acceptance, the component returns a reference that the orchestration layer can use to instantiate the workload, without exposing node identifiers or inventory.

6.2.1.2. (Optional component) QoS Estimation & Requirement Sizing Engine

This component helps the DSDT translate a QoS target (e.g., “finish simulation < 60 seconds”) into a compute requirement specification suitable for node discovery and orchestration. Unlike typical cloud autoscaling—where resource demand is inferred from continuous monitoring and reactively scaled—Digital Twin simulations often have uncertain and scenario-dependent compute cost (model complexity, solver convergence, data size, time-step resolution). The component addresses this by building predictive sizing logic that estimates “what resources are needed” *before* execution, and improves over time through feedback.

At a high level, the component supports:

- Pre-run sizing prediction: given model metadata (model type/version, parameterization, time horizon, resolution), workload context (DT class/instance features, input data volume), and QoS goal (latency/throughput/cost/energy), output a resource envelope (CPU/GPU, memory, bandwidth, expected runtime range).
- Calibration and learning loop: after each run, collect observed runtime and resource usage and update the estimator (per model version, scenario type, node capability class) to reduce uncertainty over time.
- Uncertainty-aware requests: produce requirements with confidence intervals (e.g., “likely fits in 4 vCPU/16GB, worst-case needs 8 vCPU/32GB”), enabling the node discovery broker to try conservative vs. aggressive bids.
- Policy-aware optimization: incorporate constraints such as privacy tier (PET vs non-PET), locality, and energy/performance preferences when generating requirements.

Together, these two components create an execution loop where the DSDT can (1) predict an initial sizing to meet QoS, (2) request compute through privacy-preserving bidding, and (3) continuously refine sizing accuracy using monitoring feedback—without requiring compute providers to expose their internal inventories.

6.3. Interaction with APN.

The interaction with the networking part will be abstracted and handled by DCI system.

7. Interaction with external systems and users

The DSDT functional architecture targets to facilitate the Digital Twin integration and management across systems administrated by multiple stakeholders while, at the same, transparently exploit the power of the IOWN infrastructure. *DSDT system instance*, is an instance of a DSDT FA (encompassing all or a subset of the DSDT FA functions) that is administrated by a single administrator entity. A stakeholder might be a service provider (thus, she/he will register the service through the Register interface), it might be a data provider (in this case, the non-device data will be used), it might be another Digital Twin system (thus, the Digital Twin storage will be used), or an administrator of a DSDT system instance that wish to federated with another DSDT system instance (in this case, the Federation interface will be used).

A DSDT system instance is interacting in multiple ways with external systems. This section summarizes the different interfaces used to interact with other systems (or users). Example of interactions might be with IoT systems (device data interface), with databases (non-device data), with analytics services (to be registered through the Register interface and, possibly used by the cross-domain analytics), with other Digital Twin systems (using either the Federation interface or the Digital Twin storage, or a combination of both) etc.. The figure below summarizes the available interfaces that are grouped in two: i) blue interfaces refers to DSDT function interfaces that are all described in section 4.1, ii) green interfaces are offered by IOWN Data Hub.

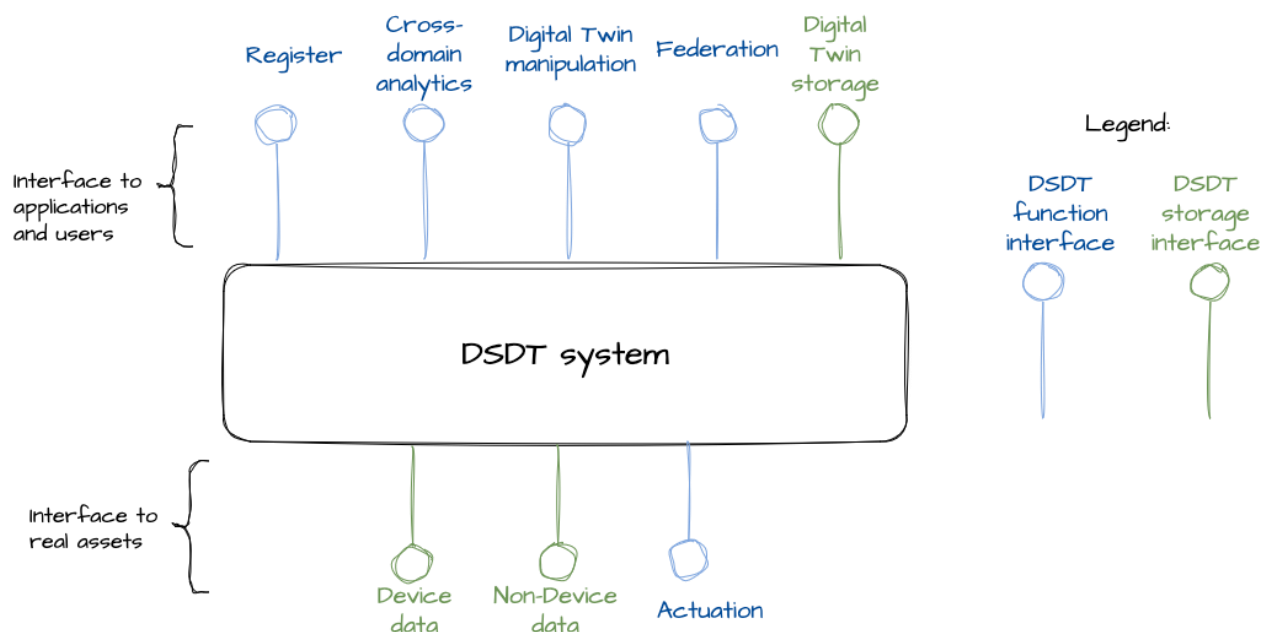


Figure 25: Exposed interfaces of the overall DSDT layer. This figures summarizes all the exposed interfaces of all the DSDT functions.

A DSDT system interacts with three different data storage systems that are:

- Device data storage handles stream data generated from electronic devices that being a physical sensor or a software component into a processing node (e.g., a computer).
- Non-device data handles static or semi-static data that is generated or update at lower frequency such as system specifications, 3D models, metadata, etc.
- Digital Twin storage handles data that is associated and contextualized to a real asset. The data is either coming from the external sources (such as the two interfaces above) or is augmented through analytics processes.

Further, a DSDT system interacts with users, applications or real assets using the following interfaces:

- *Registry function* interface exposes means to register Digital Twin components such as Digital Twin operators (see *Digital Twin analytics orchestration function*), actuation capabilities (see *Actuation function*), data sources, datasets, and visualization capabilities
- *Cross-domain analytics function* exposes XD-a-user (Cross-domain analytics user) interface that is used by a stakeholder (or its application) to request the needed data. The data might currently available as it is or it needs to go through an operator (e.g., data preparation, data aggregation or, even, ML based). To avoid ambiguities, the data requested follows a semantic approach (e.g., through an ontology). The requested data is accompanied by a description of it such as expected quality, expected Digital Twin instances, expected delays, maximum costs (e.g., in terms of energy or dataset value).
- *Digital Twin manipulation* exposes DT-m-user interface which provides the Digital Twin developer to interact with digital twin. Developer can input parameters to adjust the different simulation scenario and control the digital twin entities
- *Actuation function* uses the actuation capabilities registered through the *Registry function* to interact with actuators that enable the alteration of the real asset status reflecting the actuation command stated in the Digital Twin storage.
- *Federation function* allows another Digital Twin system instance to interact with this instance. This process encompasses the distribution, execution, aggregation, and optimization of shared models to deliver precise and holistic virtual representations. Federation across digital twins includes Federated Learning (FL), Federated Data Mining (FDM), Federated Database (FDB), and Federated Simulation (FSM).

8. Conclusion

This document has presented the second release of the Data Space for Digital Twin applications Functional Architecture (DSDT FA), providing an enhanced framework to enable and facilitate the implementation of Digital Twin applications in a multi-stakeholder data space environment. Building on Release 1, this version extends core functionalities across data handling, digital twin management, analytics, and manipulation, while introducing a comprehensive set of security functions to ensure trustworthy and resilient digital twin operations.

The DSDT FA defines a set of standardized interfaces that allow implementations to interact with real-world assets and applications, while abstracting the complexity of digital twin integration, orchestration, and federation. It also enables seamless exploitation of IOWN infrastructure capabilities, leveraging advanced communication, computing, and data technologies to support high-performance and scalable digital twin deployments.

Furthermore, the architecture is designed as a flexible and composable set of logical functions that can be adapted to diverse use cases. This release particularly strengthens support for the Network Digital Twin (NDT) as a first concrete implementation, demonstrating how the DSDT FA can address domain-specific requirements while maintaining general applicability. Overall, the DSDT FA provides a robust and extensible foundation for the evolution of digital twin applications within data spaces.

Appendix. Example instances

Smart Sustainable Mobility

Today, many cities are implementing digital twins to support urban planning, environmental management, traffic control, energy management, and other key areas. By leveraging IoT sensors to collect and monitor urban data, digital twins enable cities to make informed decisions and optimize operations through simulation.

As digital twin applications continue to evolve, they are moving beyond single-domain communication toward integration across multiple domains.

Currently, environment and mobility are major focus areas for many smart cities, and sustainable, eco-friendly mobility is becoming increasingly feasible. This example demonstrates how the digital twins of smart environment, smart driving, and smart transportation can be orchestrated to create applications that promote environmentally friendly mobility.

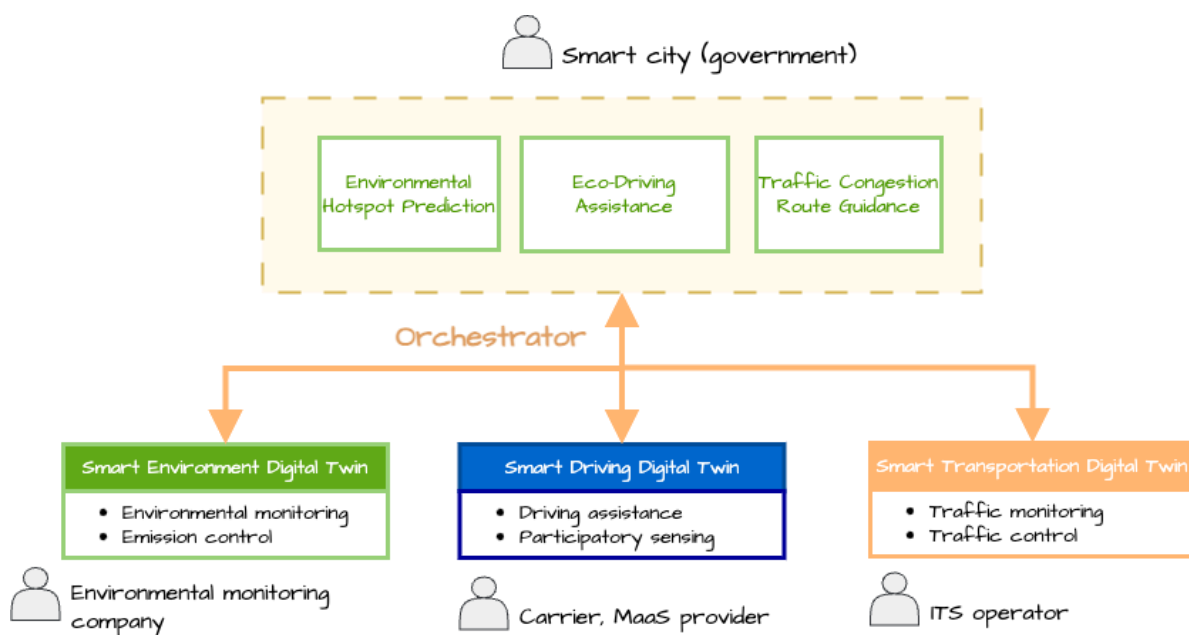


Figure A.1: Use case using the Digital Twin data handling function group.

Smart sustainable mobility scenario [Nguyen-2024] envisions a city where advanced technology optimizes transportation for environmental and social benefits. Through data-driven solutions like real-time traffic management, eco-driving assistance, and congestion pricing, this approach reduces emissions, alleviates congestion, and enhances public transit efficiency.

Digital twins orchestrate applications

- **Environmental Hotspot Prediction**

City officers are empowered to conduct detailed environmental monitoring and enforce emission restrictions by leveraging **environmental hotspot prediction**. This approach accurately forecasts air quality in high-traffic areas by utilizing data from environmental sensors mounted on vehicles, in addition to data from fixed observation stations.

In this setup, environmental hotspot prediction integrates both fixed observation stations from the **Smart Environmental Digital Twin** and mobile environmental sensors from the **Smart Driving Digital Twin**. By treating these vehicle-mounted sensors as mobile observation stations, the system enhances the observation network, enabling more precise air pollution predictions across the city.

- The **Orchestrator's Federation API** implements the Federation Function of the DSDT FA and it is used to set up access to objects within the **Smart Environment Digital Twin** and **Smart Driving Digital Twin** from the application. However, to maintain data privacy, the environmental sensor data collected within the Smart Driving Digital Twin remains restricted from external access. Instead, the air pollution prediction object in the Smart Environment Digital Twin is shared with the Smart Driving Digital Twin and configured for federated learning.
- Using the **Orchestrator's Translation API** implements the Translation Function of the DSDT FA. An application developer can use it to set up a conversion from the location and sensor values of the car objects in the Smart Driving Digital Twin to the corresponding location and observed values in the observation station objects of the Smart Environment Digital Twin. This allows the environmental data collected by the Smart Driving Digital Twin's car sensors to be seamlessly translated and integrated as observation data in the Smart Environment Digital Twin's air pollution prediction object.
- Using the **Orchestrator's Brokering API** implements the Brokering Function of the DSDT FA. An application developer can use it to receive an emission restriction plan generated by the emission simulation object within the Smart Environment Digital Twin. This emission restriction plan incorporates air pollution predictions based on environmental data collected by mobile sensors in the Smart Driving Digital Twin, effectively treating them as additional observation stations.

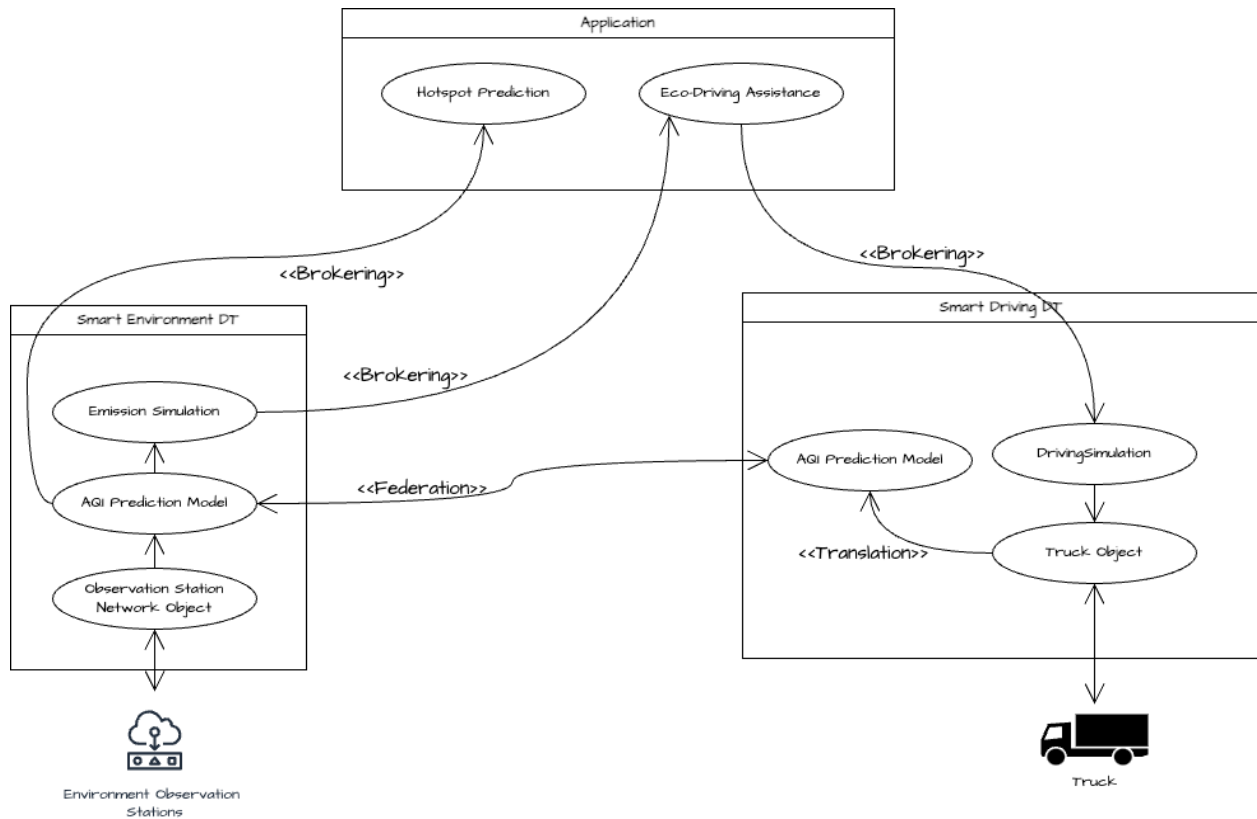


Figure A.2: Workflow for the Digital Twin application of environmental hotspot prediction and eco-driving assistance

- **Eco-driving assistance**

City officers can improve the city’s overall environmental quality through **eco-driving assistance**, which provides drivers with recommendations for environmentally friendly driving maneuvers in areas with poor air quality. This eco-driving assistance relies on the **emission restriction plan** generated by the **emission simulation** within the Smart Environment Digital Twin, which distributes emission restriction information to vehicles operating in polluted areas.

In response, the Smart Driving Digital Twin conducts a driving simulation incorporating the emission restriction data into each vehicle’s environmental profile. Based on this simulation, the navigation system guides drivers with specific maneuvers designed to control emissions.

Additionally, when hotspot prediction is combined with eco-driving assistance, the Smart Environment Digital Twin benefits from enhanced air pollution predictions, as it receives supplemental environmental data from car-based sensors. This integration enables more detailed emission simulations and enhances the effectiveness of eco-driving assistance based on emission restriction plans, helping further reduce urban pollution levels.

- The Orchestrator's Federation API configures the application to access objects within both the Smart Environment Digital Twin and the Smart Driving Digital Twin. It also sets up the air pollution prediction object in the Smart Environment Digital Twin to be shared with the Smart Driving Digital Twin, facilitating federated learning across these systems.
- Through the Orchestrator's Brokering API, the application can receive the emission restriction plan generated by the emission simulation object in the Smart Environment Digital Twin. The application then uses this plan to determine specific emission restrictions for vehicles within targeted areas. Using the Orchestrator's Brokering API, these restrictions are sent directly to the Smart Driving Digital Twin instances corresponding to the affected vehicles.
- The Orchestrator's Translation API enables the application to interpret the environmental sensor data attached to each vehicle in the Smart Driving Digital Twin as observed data within the Smart Environment Digital Twin's observation station objects. This translated data is then input into the air pollution prediction object, enhancing the accuracy of environmental forecasts and supporting proactive pollution management.
- **Traffic congestion reduction route guidance**

City officers can mitigate severe traffic congestion by recommending alternative routes and implementing congestion pricing strategies to evenly distribute traffic and prevent vehicle concentration on specific roads. For congestion reduction guidance, the application synchronizes each vehicle's position in the Smart Transportation Digital Twin with the corresponding car in the Smart Driving Digital Twin. This enables recommended routes to be sent to vehicles approaching areas where congestion reduction is needed. Additionally, based on the traffic flow plan generated by the traffic simulation in the Smart Transportation Digital Twin, the recommended route and congestion pricing information are sent to cars in the Smart Driving Digital Twin traveling on roads targeted for congestion management. Upon receiving this information, the Smart Driving Digital Twin performs a driving simulation to identify an optimal route for its vehicle and sends this route guidance to the vehicle's navigation system.

 - Use the Orchestrator's Synchronization API to set up synchronization between the vehicle object recognized by the road object in the Smart Transportation Digital Twin and the car object in the Smart Driving Digital Twin. Since vehicles on the road change from time to time, the vehicle license plate number recognized by the road object and the license plate number of the car object should be identical for synchronization.
 - The Orchestrator's Brokering API is used to enable the application to receive the traffic flow plan generated by the traffic simulation object of the Smart Transportation Digital Twin. The application determines from the traffic flow plan the recommended route for the vehicle on the target road, and configures the Brokering API

to send them to the Smart Driving Digital Twin. In doing so, it identifies as the destination the Smart Driving Digital Twin of the car that has been synchronized with the target vehicle by the Synchronization API.

References

[ETSI-GS-CIM-009] ETSI ISG CIM, Context Information Management (CIM); NGSI-LD API, V1.8.1, March 2024

[IOWN-GF-Vision-2030] IOWN GF, Innovative Optical and Wireless Network Global Forum Vision 2030 and Technical Directions, version 2, 2022, https://iowngf.org/wp-content/uploads/2023/03/IOWN_GF_WP_Vision_2030_2.0-2.pdf

[IOWN-GF-DCI] IOWN GF, Data-Centric Infrastructure Functional Architecture, version 2.0, March 2023, https://iowngf.org/wp-content/uploads/2023/04/IOWN-GF-RD-DCI_Functional_Architecture-2.0.pdf

[IOWN-GF-DSDT-FA-rel1] Data Space for Digital Twin Applications – Functional Architecture, version 1.0, December 2024, https://iowngf.org/wp-content/uploads/2025/02/IOWN-GF-RD-DSDT_Functional_Architecture-1.0.pdf

[IOWN-GF-DTF-AR] IOWN GF, Digital Twin Framework Analysis Report, Version 1.0, 2022, https://iowngf.org/wp-content/uploads/formidable/21/Digital_Twin_Framework_Analysis_Report.pdf

[IOWN-GF-IDH] IOWN GF, Data Hub Functional Architecture, version 2, March 2023, https://iowngf.org/wp-content/uploads/formidable/21/IOWN-GF-RD-Data_Hub_Functional_Architecture-2.0.pdf

[IOWN-GF-DTC] IOWN GF, Digital Twin computing, version 2, October 2019, https://iowngf.org/wp-content/uploads/formidable/21/DTC_Whitepaper_en_2_0_0-1.pdf

[IOWN-GF-NDT] IOWN GF Network Digital Twin use case document, version 2, July 2024, https://iowngf.org/wp-content/uploads/formidable/21/IOWN-GF-RD-NDT_Use_Case-2.0.pdf

[IOWN-GF-NDT-REQ] IOWN GF Network Digital Twin – Use Case and Technical Requirements, December 2025, <https://iowngf.org/wp-content/uploads/2025/12/IOWN-GF-RD-NDT-Use-Case-Tech-Reqs-1.0.pdf>

[IOWN-GF-MFS] Functional Architecture for Protection of Data in Motion: Multi-Factor Security Key Exchange and Management, Version 1.0, October 2024, https://iowngf.org/wp-content/uploads/formidable/21/IOWN-GF-RD-MFS_Functional_Architecture-1.0.pdf

[IOWN-GF-PET] Functional Architecture for Protection of Data in Use: IOWN Privacy Enhancing Technologies, Version 1.0, October 2024, https://iowngf.org/wp-content/uploads/formidable/21/IOWN-GF-RD-PETs_Functional_Architecture-1.0.pdf

[Nguyen-2024] Nguyen, Do-Van, Minh-Son Dao, and Koji Zettsu. "Digital Twin Orchestration: Framework and Smart City Applications." (2024)

Abbreviations and acronyms

AI	Artificial Intelligence
APN	All-Photonics Network
CPS	Cyber-Physical System
DCI	Data-Centric Infrastructure
DSDT	Data Space for Digital Twin applications
DT	Digital Twin
DTF	Digital Twin Framework
FDB	Federated Database
FDM	Federated Data Mining
FL	Federated Learning
FSM	Federated Simulation
gNMI	gRPC Network Management Interface
gRPC	Google Remote Procedure Call
IDH	IOWN Data Hub
IOWN	Innovative Optical and Wireless Network
JSON	JavaScript Object Notation
JSON-LD	JavaScript Object Notation for Linked Data
NDT	Network Digital Twin
NGSI-LD	Next Generation Service Interfaces - Linked Data
OWL	Web Ontology Language
PCE	Path Computation Element
PET	Privacy-Enhancing Technologies
QoS	Quality of Service
RDF	Resource Description Framework
TAPI	Transport API

TEE Trusted Execution Environment

YANG Yet Another Next Generation

History

Revision	Release Date	Summary of Changes
1	December 2024	First DSDT Functional Architecture
2	June 2026	The Translation function (Section 4.2.1), Actuation function (Section 4.2.1), Analytics Orchestration function (Section 4.2.2), and Visualization workflow (Section 4.2.3) have been evolved, and a new Analytics Monitoring function (Section 4.2.2) has been defined to support the Network Digital Twin use case [IOWN-GF-NDT-REQ]. Security functions have been integrated into the DSDT FA. A new section on the Added Value perspective has been included in this document.