CATENA-X

STANDARD

CX - 0002 DIGITAL TWINS IN CATENA-X v.2.1.0

Contact: standardisierung@catena-x.net

Table of Contents

ABSTRACT

The Catena-X network is about accessing/sharing/providing/using data, formulated in the different use cases. This standardization scenario is about how the data, and the data models look like and how the modelling has to be done, so that data between ecosystem partners can be shared, lossless and in a machine-readable way. This document focuses on Digital Twins and their application and administration within Catena-X.

The purpose of this standard is to provide concepts and specifications in order to allow proper data provisioning with Digital Twins in Catena-X.

FOR WHOM IS THE STANDARD DESIGNED

This standard is designed as an implementable specification and thus is relevant for all technical roles concerned with APIs and Data Models in the Catena-X network

COMPARISON WITH THE PREVIOUS VERSION OF THE STANDARD

Adjustment to new template
More precise statements on registration in the EDC.
More precise statements on registration of submodels in the Digital Twin Registry.

1 INTRODUCTION

1.1 AUDIENCE & SCOPE

This standard is relevant for

data provider / consumer
solution provider

The standard is applicable in the following cases for the following roles:

all data providers who need to provide information via Digital Twins
all data consumers and business application provider who need access to data provided via Digital Twins
solution providers of a Digital Twin Registry
onboarding service providers who need to offer core service of a Digital Twin Registry to their customers
enabling service providers who need access to data provided via Digital Twins
consulting service providers who need to explain how Digital Twins are implemented and/or used

1.2 CONTEXT AND ARCHITECTURE FIT

> *This section is non-normative*

Catena-X creates a central, uniform, and consistent solution for data share in automotive industry. In this context, the exchange of data is an essential requirement for the success of this network. For this purpose, Catena-X provides various methods, tools, and standards

to ensure semantic interoperability. Digital Twins have established themselves here as a central element for structuring and accessing data. With the help of defined semantics, both data provision and app development are simplified and encouraged.

1.2.1 Digital Twins in Catena-X

The term Digital Twin (DT) describes a digital representation of an asset sufficient to meet the requirements of a set of use cases.

Any asset - it can be an actual physical asset like a drilling machine but also something virtual like a web service - has a digital representation with consistent semantics. Hence, Digital Twins adhere to the following characteristics:

- The DT has at least one Catena-X-wide unique identifier (ID).
- An asset can have more than one DT.
- DTs are organized by a set of Aspects. This set can be extended over lifetime.
- An Aspect of a DT includes both structural as well as behavioral data and models, including operations and simulation models.
- The semantics of an Aspect can be described via semantic models.
- A single Aspect can be connected to different heterogeneous data sources, including behavioral models.
- The DT can represent type assets (*e.g.*, virtual prototype of a car) and instance assets (*e.g.*, real car).
- A DT can cover the whole asset lifecycle including, *e.g.*, the planning, production, sales, use, and decommissioning phases. However, in practice there may be more than one twin with different IDs representing different lifecycle phases, *e.g.*, one twin for types and multiple twins for instances.
- The DT represents current available information about an asset, synchronized at a specified frequency and fidelity, which can be leveraged for simulation and business process integration.
- By using Aspects, a DT can reference other DTs to express "part of" or "consists of" relations.

1.2.2 Digital Twin Registry

A Digital Twin Registry (DTR) is an operated solution which lists Digital Twins and their respective Aspects. Each Digital Twin represents a single asset. Some basic information about the asset being represented is part of each entry in a DTR.

For each asset several data sets in form of Aspects can be provided. These Aspects are referenced within each Digital Twin together with some access information, the Aspect endpoints.

Moreover, a DTR also offers basic discovery functionality to find Digital Twin(s) representing an asset under consideration.

In general, every data provider in the dataspace must decide how and where to operate a DTR.

The data provider needs to register all its Digital Twins including its respective Aspects to its DTR service in order to reveal its "offer" of sharing respective data sets.

The data offered by a Digital Twin via Submodels should be semantically described by a semantic Aspect meta model conformant to CX-0003.

1.2.3 Asset Administration Shell

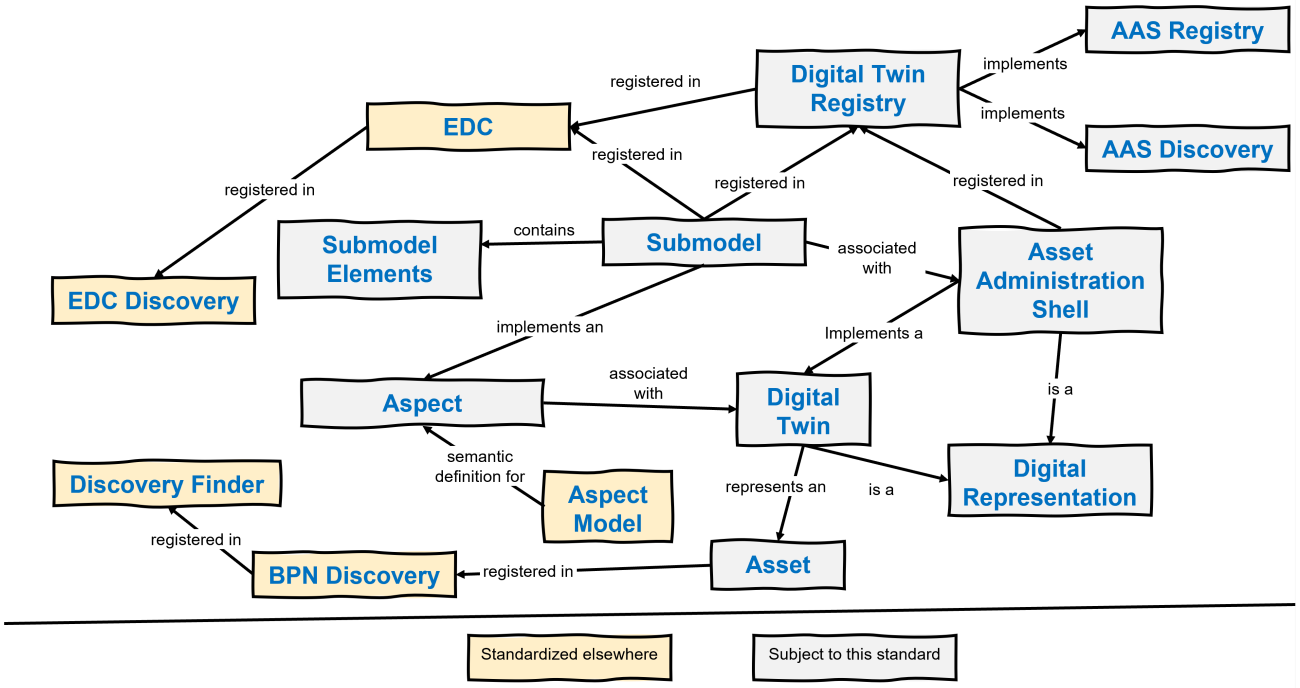
The Asset Administration Shell (AAS) is a key concept of Industry 4.0 (or "Industrie 4.0" in German), maintained by the Industrial Digital Twin Association (IDTA), and is used to describe an asset electronically in a standardized manner. The AAS is a standardized way to implement a Digital Twin. One of the main concepts of the AAS is the concept of Submodels, each of which can characterize the asset by describing its Aspects for different use cases and data consumers.

The AAS standardized a set of API methods and resources to access data of a Digital Twin.

Also an Asset Administration Shell Registry service and other services in the context of Digital Twins are standardized.

In Catena-X the semantics of a Submodel is described via an Aspect Model conformant to standard CX-0003, preferrable by using standardized properties conformant to standard CX-0044.

The following figure gives a high-level overview how the concepts relevant for this standard relate with each other and concepts from neighboring domains.



In general, the AAS has proven to be suitable for the following:

> how to represent data exchanged in a standardized way between two parties (API payload)
>
> how to access data exchanged in a standardized way between two parties (API operations)
>
> how to find data available for the asset under consideration for data exchange between two parties (Digital Twin Registry) in a standardized way

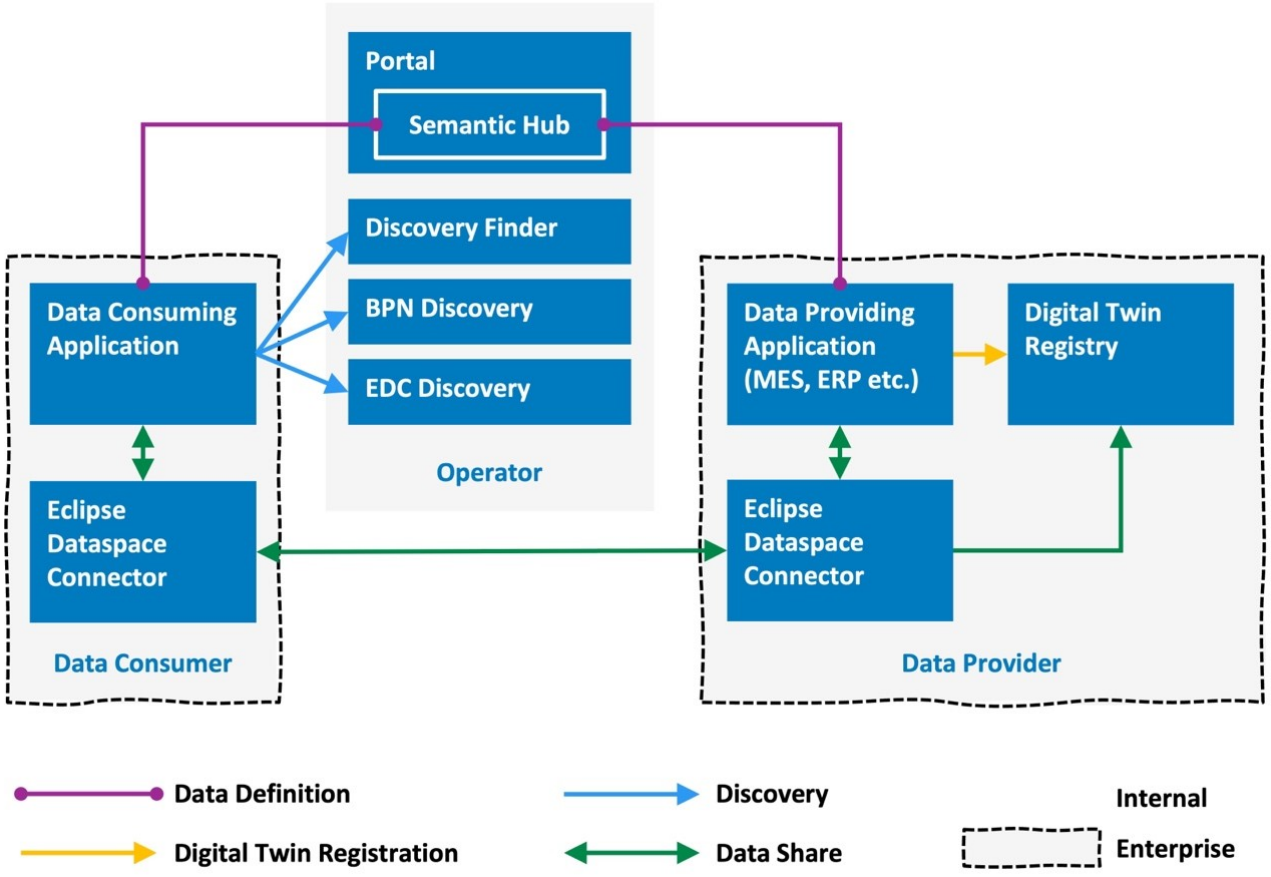
The Asset Administration Shell Reading Guide gives an overview for different stakeholders. This reading guide together with detailed technical documentation can be found in the Content Hub of the IDTA and on GitHub: https://github.com/admin-shell-io/aas-specs.

1.2.4 Architecture Overview

In the following the architecture is sketched. The Digital Twin Registry (DTR) component is a non-centralized component. Typically, each data provider offers its own DTR, either using an enablement service provider that also operates the DTR for the data provider or operating it itself.

The DTR does not only contain pure registration functionality but also basic discovery functionality based on asset identifiers. The corresponding APIs for this kind of discovery are specified in this document.

A DTR is accessed via a dataspace connector (EDC) conformant to standard CX-0018. Business solutions first need to find the relevant EDC(s) and thus the relevant DTR. Beside EDC discovery (see standard CX-0001) additional discovery services (see standard CX-0053) are provided to reduce the number of dataspace connectors that need to be accessed by the business application.



1.3 CONFORMANCE AND PROOF OF CONFORMITY

> *This section is non-normative*

As well as sections marked as non-normative, all authoring guidelines, diagrams, examples, and notes in this specification are non-normative. Everything else in this specification is normative.

The key words MAY, MUST, MUST NOT, OPTIONAL, RECOMMENDED, REQUIRED, SHOULD and SHOULD NOT in this document are to be interpreted as described in [BCP 14], [RFC2119], [RFC8174] when, and only when, they appear in all capitals, as shown here.

1.5 Proof of conformity

> *This section is non-normative*

All participants and their solutions will need to prove that they conform to the Catena-X standards. To validate that the standards are applied correctly, Catena-X employs Conformity Assessment Bodies (CABs).

1.5.1 Proof of Conformity for Digital Twin Registry Solutions

A Digital Twin Registry solution MUST provide http/REST APIs conformant to the openAPI specification as defined in this document.

In case the Digital Twin Registry solution already has a valid certificate of the Industrial Digital Twin Association (IDTA) including the required service specification profiles the simplified certification process of Catena-X e.V. holds.

If there is no valid certificate available from IDTA then Digital Twin Registry solution providers MUST prove their conformity by providing:

An openAPI specification of the endpoints
API Response of the implementation MUST match to the response structure of the required API specifications in this document

A Digital Twin Registry Solution MUST include mechanisms that allow to ensure confidentiality and integrity of data, and compliance with antitrust laws.

On default, the read access to Digital Twins SHOULD be enabled by Digital Twin Registry Solutions to data providers only.

1.5.2 Proof of Conformity for Data Providers

A data provider MUST offer the http/REST APIs for its Digital Twin Registry service conformant to this specification.

In case the Digital Twin Registry solution already has a valid certificate of the Industrial Digital Twin Association (IDTA) including the required service specification profiles the simplified certification process of Catena-X e.V. holds.

The Digital Twin Registry service used by the data provider MUST be registered in the EDC selected by the data provider to provide access to its Digital Twin Registry.

A data provider MAY create and register Digital Twins using the http/REST APIs conformant to the openAPI specification as defined in this document.

The data provider MUST offer the READ operations for Digital Twins and its Aspects conformant to this specification.

The endpoints offered by the data provider MUST be made accessible via EDC as specified in this document or other use case related standards.

Appropriate usage and access policies conformant to standard CX-0018 MUST be defined for accessing the Digital Twin Registry itself as well as for the Submodels access is granted.

Data providers MUST comply with antitrust law, *i.e.*, competitively sensitive information (*e.g.* customer names, supplier names, prices, price models, internal knowhow, sales and/or purchasing strategies) MUST NOT be published via a DTR.

The data provider SHOULD use the unique identifier of the standardized Aspect Model conformant to CX-0003 when registering a new Submodel endpoint to a DTR.

A data provider SHOULD add the required specific asset IDs for each Digital Twin depending on the use case the data provider is involved in.

A data provider SHOULD add information to available discovery services conformant to standard CX-0001 and CX-0053 - if available - to enable data consumers to find the EDC and thus the Digital Twin Registry the data consumer is interested in.

1.5.3 Proof of Conformity for Data Consumers

A data consumer, business application provider or enabling service provider MAY lookup the endpoints of the Submodels relevant for the use case using the http/REST APIs conformant to the openAPI specification as defined in this document.

Since there are several Digital Twin Registries in the dataspace data consumers, business application providers or enabling service providers MAY first lookup the available Digital Twin Registry endpoints of the relevant EDCs using the corresponding standardized EDC lookup and discovery services (see standard CX-0001).

Additionally, data consumers MAY use standardized discovery services - if available -, *e.g.*, to find the EDC for a specific company via its BPN (see standard CX-0053).

1.6 Examples

Examples can be found in the Tractus-X DTR's documentation and the Digital Twin Kit

1.7 Terminology

> *This section is non-normative*

Aspect

> *a domain-specific view on information and functionality associated with a specific Digital Twin with a reference to a concrete Aspect Model.*

Note 1 to entry: An Aspect is a software service to retrieve the actual runtime data of a Digital Twin (current or aggregated) from a data source or to trigger operations. Thus, an Aspect is built with an implementation that ensures that the exchanged data is compliant to the specification of the referenced Aspect Model via a defined interface.

Note 2 to entry: Aspects are registered (incl. their "API endpoint" information) within the Digital Twin to which they belong in the Digital Twin Registry.

Note 3 to entry: an Aspect corresponds to a *Submodel* in the *Asset Administration Shell*

[SOURCE: Eclipse Semantic Modeling Framework (ESMF), editorial changes and notes added]

Aspect Model

> *a formal, machine-readable semantic description (expressed with RDF/turtle) of data accessible from an Aspect.*

Note 1 to entry: An Aspect Model must adhere to the Semantic Aspect Meta Model (SAMM), *i.e.*, it utilizes elements and relations defined in the SAMM and is compliant with the validity rules defined by the SAMM.

Note 2 to entry: Aspect Models are logical data models which can be used to detail a conceptual model in order to describe the semantics of runtime data related to a concept. Further, elements of an Aspect Model can/should refer to terms of a standardized Business Glossary (if existing).

Note 3 to entry: An Aspect Model may describe the semantics of a *Submodel*.

[SOURCE: Eclipse Semantic Modeling Framework (ESMF), editorial changes and notes added]

Asset Administration Shell

> *standardized digital representation of an asset*

Note 1 to entry: Asset Administration Shell and Administration Shell are used synonymously.

[SOURCE: IEC 63278-1, note added]

Digital Twin

> *digital representation, sufficient to meet the requirements of a set of use cases*

Note 1 to entry: in this context, the entity in the definition of digital representation is typically an asset.

[SOURCE: IIC Vocabulary IIC:IIVOC:V2.3:20201025, adapted (an asset, process, or system was changed to an asset)]

Digital Representation

> *information and services representing an entity from a given*

EXAMPLE 1: examples of information are properties (*e.g.*, maximum temperature), actual parameters (*e.g.*, actual velocity), events (*e.g.*, notification of status change), schematics (electrical), and visualization information (2D and 3D drawings).

EXAMPLE 2: examples of services are providing the history of the configuration data, providing the actual velocity, and providing a simulation.

EXAMPLE 3: examples of viewpoints are mechanical, electrical, or commercial characteristics.

[SOURCE: IEC 63278-1, editorial changes]

Submodel

> *container of SubmodelElements defining a hierarchical structure consisting of SubmodelElements*

[SOURCE: IEC 63278-1]

Submodel Element

> *elements in a Submodel*

[SOURCE: IEC 63278-1]

2. Digital Twin Registry API for Solution Providers [NORMATIVE]

2.1 API Specification

2.1.1 Standards and Profiles

The specification *Specification of the Asset Administration Shell - Part 2: Application Programming Interfaces* is the basis for the Digital Twin Registry implementation in Catena-X. The document can be found in the content hub of IDTA: https://industrialdigitaltwin.org/content-hub.

The API is offered as swagger implementation in addition to its formal specification published on https://industrialdigitaltwin.org/.

For relevant profiles of the Service Specifications see chapters on *API endpoints & resources*.

2.1.2 API Endpoints & resources

The API MUST be implemented as specified for the profiles:

> *Asset Administration Shell Registry Service Specification V3.0 READ Profile SSP-002*
>
> *Discovery Service Specification V3.0 Profile SSP-001*

The following profile SHOULD be implemented:

> *Asset Administration Shell Registry Service Specification V3.0 Profile SSP-001*

The following deviations are defined:

> Length of *ProtocolInformation/subprotocolBody* string MUST be 2048
> DELETE and POST operations of Discovery Interface are optional to be implemented

EXAMPLE for Self-Description (**GetSelfDescription**) of a Digital Twin Registry solution :

```
{
  "profiles": [
    "https://admin-shell.io/aas/API/3/0/DiscoveryServiceSpecification/SSP-001",
    "https://admin-
shell.io/aas/API/3/0/AssetAdministrationShellRegistryServiceSpecification/SSP-002"
```

```
    ]
}
```

2.1.3 Available Data Types

The API MUST use JSON as the payload transported via HTTP.

For explanation of data types see *Specification of the Asset Administration Shell - Part 2: Application Programming Interfaces*.

2.1.4 EDC Data Asset Structure

Not applicable, since EDC data asset(s) are created by data provider not by DTR solution provider.

2.1.5 Error Handling

Error response **501 Not Implemented** MUST be used for operations or parameter values not yet supported.

For other error codes and error handling see *Specification of the Asset Administration Shell - Part 2: Application Programming Interfaces*.

3. Digital Twin Registry API for Data Providers [NORMATIVE]

3.1 API Specification

3.1.1 Standards and Profiles

The specification of the Asset Administration Shell - Part 2: Application Programming Interfaces is the basis for the Digital Twin Registry implementation in Catena-X.

The API is offered as swagger implementation in addition to its formal specification published on https://industrialdigitaltwin.org/.

For relevant profiles of the Service Specifications see chapters on *API endpoints & resources*.

3.1.2 API Endpoints & resources

The API MUST be implemented as specified for the profiles:

> Asset Administration Shell Registry Service V3.0 READ Profile SSP-002
>
> Discovery Service Specification V3.0 Profile SSP-001

The WRITE operations of the following profile MAY be used to create and delete Digital Twins:

> Asset Administration Shell Registry Service Profile V3.0 Profile SSP-001

The same deviations are defined as for Digital Twin solution providers.

3.1.3 Available Data Types

The API MUST use JSON as payload transported via HTTP.

For explanation of data types see Specification of the Asset Administration Shell - Part 2: Application Programming Interfaces.

3.1.4 EDC Data Asset Structure

A Digital Twin Registry that is made accessible via an EDC MUST be registered with the following values for given asset properties.

1. `"asset:prop:type": "data.core.digitalTwinRegistry"`,

2. `"dct:type": {"@id":"https://w3id.org/catenax/taxonomy#DigitalTwinRegistry"}`,
3. `"cx-common:version": "3.0"` . Please note that the "cx-common" prefix must be registered to "https://w3id.org/catenax/ontology/common#" in the @context object when creating an Asset.

For more details on EDC see CX-0018.

3.1.5 Error Handling

Error response **501 Not Implemented** MUST be used for operations or parameter values not yet supported.

For other error codes and error handling see *Specification of the Asset Administration Shell - Part 2: Application Programming Interfaces*.

4. Submodel API for Data Providers [NORMATIVE]

4.1 API Specification

4.1.1 Standards and Profiles

The Specification of the Asset Administration Shell - Part 2: Application Programming Interfaces is the basis for exchanging data via Digital Twins in Catena-X.

The API is offered as swagger implementation in addition to its formal specification published on https://industrialdigitaltwin.org/.

For relevant profiles of the Service Specifications see chapters on *API endpoints & resources*.

4.1.2 API Endpoints & resources

A Submodel that provides data MUST be implemented in conformance to the API-definition of Submodel Service Specification V3.0 VALUE ONLY Profile SSP-003 of the Submodel Service Specification.

The following additional restrictions apply:

The `semanticId` of a referred Submodel MUST also be added to the Submodel Descriptor registered for the DT (*SubmodelDescriptor/semanticId*).

A data provider MUST use the unique identifier of the Aspect Model standardized in Catena-X conformant to standard CX-0003 when registering a corresponding new Aspect to a Digital Twin (*Submodel/semanticId*).

The `subprotocol` must be set to "DSP" (*SubmodelDescriptor/endpoints/protocolInformation/subprotocol*).
The subprotocolBody must be set according to the concatenation of the following key-value-pairs (assigned by a "=" and separated by a semicolon ";"):

`id` represents the id of that dcat:DataSet in the Data Providers catalog that contains the Submodel.
`dspEndpoint` represents the endpoint of the Data Provider's Control Plane where the catalog containing the relevant dcat:DataSet is located.

A full example satisfying these restrictions is at the end of this chapter.

Beside Aspect Models standardized in Catena-X also other Aspects may be registered, either conformant to proprietary standards or standards from other organizations. However, these Aspects are not fit to be used in Catena-X use cases or standards. In either case, an Aspect Model conformant to standard CX-0003 SHOULD be made available for these Aspects.

The following deviations are allowed, *i.e.*, the following API operations and operation parameters SHOULD be supported but these are not mandatory to be implemented.

provision of API /**description**

support of API operation /**submodel** (*i.e.*, logical interface operation "GetSubmodel with logical parameter "Content"="Normal", the API operation /**submodel**/**$value** extending the default operation's URL with the parameter /**submodel**/**$value** remains mandatory)
support of API operation /**submodel**/**submodel-elements**/< **IdShortPath** >/**invoke** and /**submodel**/**submodel-elements**/< **IdShortPath** > /**invoke**/**$value**
support of query parameter **Extent=WithoutBLOBValue**

NOTE for Profile SSP-003 of the Submodel Service Specification the following parameters are not included:

no provision of API /**serialization**
no support of logical parameter **Content=Metadata**, *i.e.*, no support of API operation with path parameter "$metamodel" /**submodel$metamodel**
no support of logical parameter **Content=Reference**, , *i.e.*, no support of API operation with path parameter "$reference" /**submodel**/**$reference**
no support of logical parameter **Content=Path**, *i.e.*, no support of API operation with path parameter "$path" /**submodel**/**$path**
no support of query parameter **Level=Core** (only of **Level=Deep**)

EXAMPLE for a Submodel descriptor in the DTR accessible via an EDC:


```
{
  "id": "<unique ID of Submodel>",
  "semanticId": {
    "type": "ExternalReference",
    "keys": [
      {
        "type": "GlobalReference",
        "value": "urn:bamm:io.catenax.material_for_recycling:1.1.0#MaterialForRecycling"
      }
    ]
  },
  "endpoints": [
    {
      "protocolInformation": {
        "href": "https://edc.data.plane/<path>/api/v3.0/submodel",
        "endpointProtocol": "HTTP",
        "endpointProtocolVersion": [
          "1.1"
        ],
        "subprotocol": "DSP",
        "subprotocolBody": "id=123;dspEndpoint=http://edc.control.plane/",
        "subprotocolBodyEncoding": "plain",
        "securityAttributes": [
          {
            "type": "NONE",
            "key": "NONE",
            "value": "NONE"
          }
        ]
      },
      "interface": "SUBMODEL-3.0"
```

```
      }
    ]
}
```

4.1.3 Available Data Types

The API MUST use JSON as payload transported via HTTP.

4.1.4 EDC Data Asset of Submodels registered in Digital Twin Registry

Access to the Submodels of a Digital Twin MUST take into account EDC access and usage policies defined for them (see standard CX-0018).

The data provider MAY cluster several Submodels into one EDC asset.

If a single Submodel is registered as a single Data Asset at the EDC, its MUST be registered with the following values for given asset properties.

1. `"dct:type": {"@id":"https://w3id.org/catenax/taxonomy#Submodel"}`,
2. `"cx-common:version": "3.0"`. Please note that the "cx-common" prefix must be registered to "https://w3id.org/catenax/ontology/common#" in the @context object when creating an Asset.

In that case, Data Providers SHOULD also add a property `aas-semantics:semanticId` that is set to the composite semanticId of the Submodel that the Asset represents. Please note that the "aas-semantics" prefix must be registered to "https://admin-shell.io/aas/3/0/HasSemantics/" in the @context object when creating an Asset.

For more details on EDC see CX-0018.

4.1.5 Error Handling

Error response **501 Not Implemented** MUST be used for API operations and parameter values not yet supported.

For error handling see Specification of the Asset Administration Shell - Part 2: Application Programming Interfaces.

5. Submodel API for Data Consumers [NORMATIVE]

The specification Specification of the Asset Administration Shell - Part 2: Application Programming Interfaces is the basis for consuming data via Digital Twins in Catena-X.

The API is offered as swagger implementation in addition to its formal specification published on https://industrialdigitaltwin.org/.

This is the relevant service specification profile for data consumers:

> Submodel Service Specification V3.0 READ Profile SSP-003

The READ API operations of the Asset Administration Shell profile SSP-003 of the Submodel Service Specification MAY be used to access the Submodels provided by data providers.

The logical parameter "Content" is realized via path suffixes (starting with $) like in */submodel/$value*. The endpoint within the Digital Twin Registry is not including the path suffixes. This is why the path suffix needs to be explicitly added to the endpoint before calling the value-only Submodel operation.

NOTE The logical *GetSubmodel* operation is not called explicitly by the data consumer. Instead the endpoint as provided via the Digital Twin Registry for the asset and *Submodel* of interest is called with GET. Additionally, the data consumer MAY need to set parameters

before calling the API operation.

NOTE The logical parameter "Content" is realized via path suffixes (starting with $) like in */submodel/$value*. The reason is that the payload differs depending on the value of the parameter. A logical parameter like "Level" is realized as query parameter.

NOTE The logical operation *GetSubmodel* can be implemented in different ways. The only relevant information for the data consumer is the endpoint information in the Digital Twin Registry. Besides its availability in the Submodel Service Specification the operation is also available in the Asset Administration Shell Service Specification as well as the Submodel Repository Service Specification or Asset Administration Shell Repository Service Specification via superpaths.

---

6. References

6.1 Normative References

Specification of the Asset Administration Shell - Part 1: Metamodel. V3.0, April 2023, IDTA number: 01001-3-0 On IDTA Content Hub

Specification of the Asset Administration Shell - Part 2: Application Programming Interfaces. V3.0, April 2023, IDTA number: 01002-03-1 On IDTA Content Hub

6.2 Non-Normative References

> *This section is non-normative*

Asset Administration Shell Reading Guide

CX-0003 SEMANTIC ASPECT META MODEL. In Catena-X Standard Library

CX-0018 ECLIPSE DATASPACE CONNECTOR (EDC). In Catena-X Standard Library

CX-0001 EDC DISCOVERY API. In Catena-X Standard Library

CX-0053 BPN Discovery Service APIs. In Catena-X Standard Library

CX-0044 ECLASS. In Catena-X Standard Library

6.3 Reference Implementations

> *This section is non-normative*

The following open-source project implements a Digital Twin Registry solution conformant to this standard:

https://github.com/eclipse-tractusx/sldt-digital-twin-registry

ANNEXES

FIGURES

> *This section is non-normative*

```
[OPTIONAL] Add figures here if necessary. Please delete if no figures are provided
```

TABLES

> *This section is non-normative*

```
[OPTIONAL] Add Tables here here if necessary. Please delete if no tables are provided
```

ABOUT THIS DOCUMENT & MOTIVATION

DISCLAIMER & LIABILITY

REVISIONS & UPDATE

COPYRIGHT & TRADEMARKS