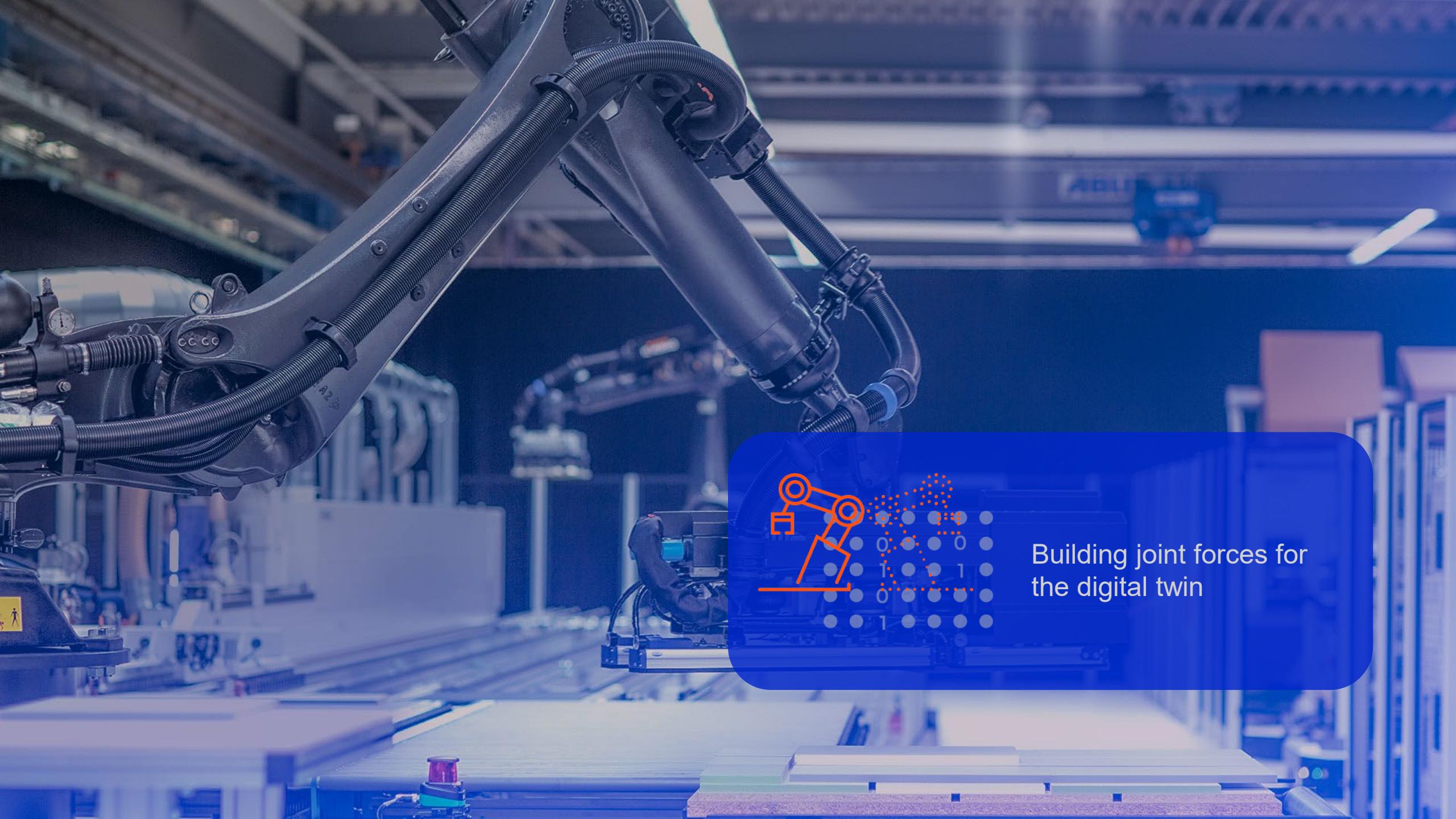


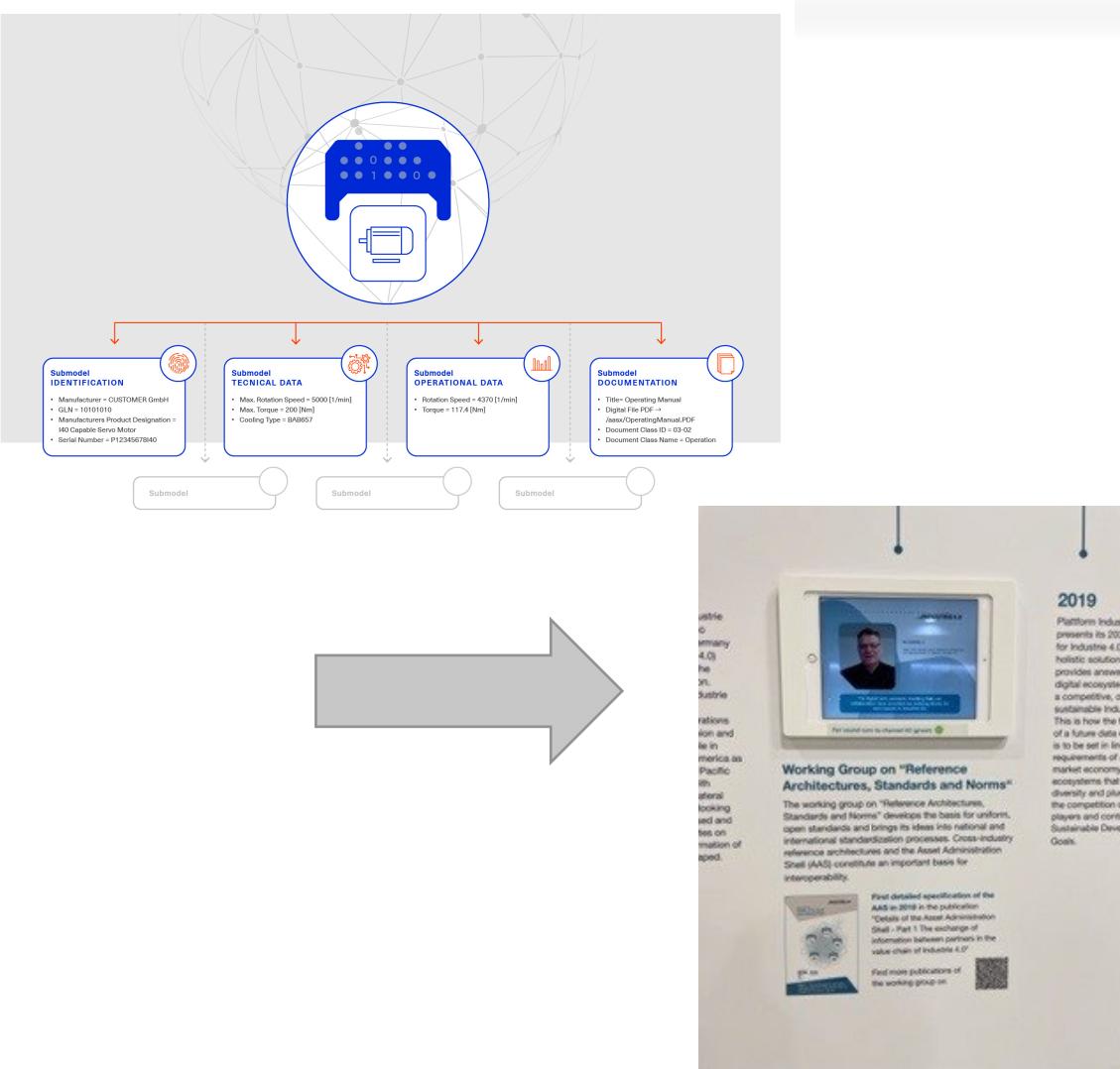
Tutorial

Specification Asset Administration Shell
Part 1 Metamodel v3.0 April 2023



Building joint forces for
the digital twin

From Concept to Specification



2018
V1.0

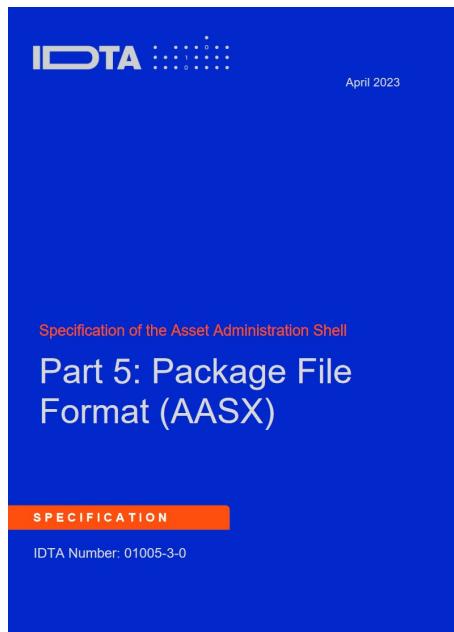
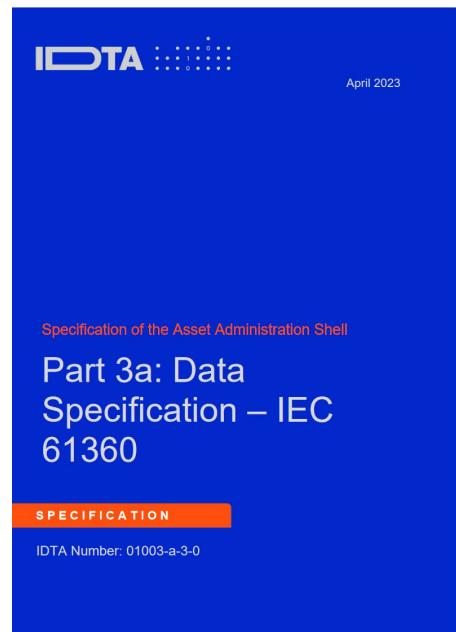
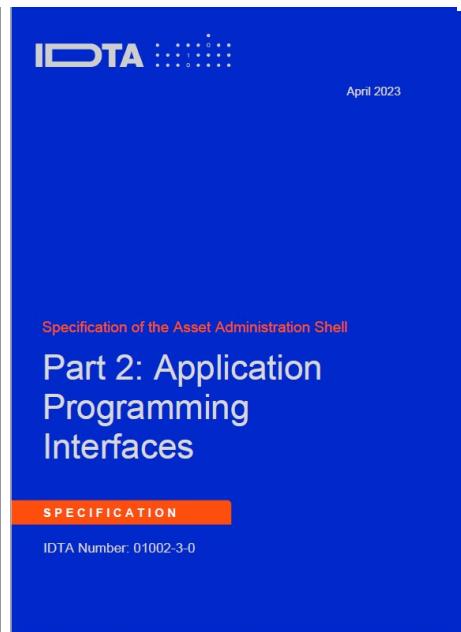
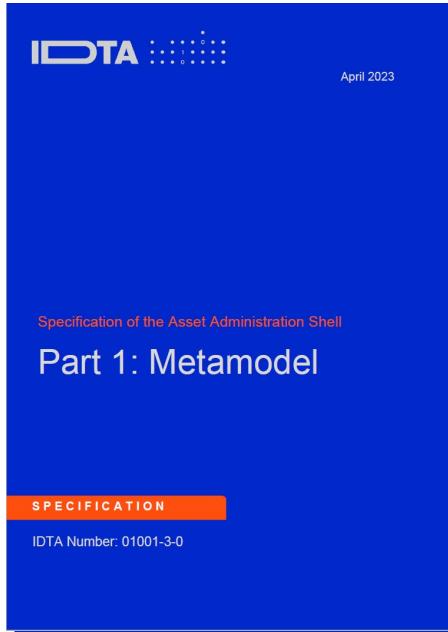
April 2023
V3.0



Specifications



IDTA



V3.0
April 2023



IEC 63278



This Tutorial is
about Part 1

For whom is this tutorial?

- You should have an idea of the benefits of introducing digital twins to your domain
- You should know about the basic concepts of the Asset Administration Shell
- You should have basic knowledge in UML modeling
- You are an architect and want to learn more about the underlying information model of the Asset Administration Shell
- You are a developer and want to upgrade to the new version or start your first implementation

5 The Information Metamodel of the Asset Administration Shell (normative)
5.1 Introduction
5.2 Overview Metamodel of the Asset Administration Shell
5.3 Metamodel Specification Details: Designators
5.3.1 General
5.3.2 Common Attributes
5.3.3 Asset Administration Shell Attributes
5.3.4 Asset Information Attributes
5.3.5 Submodel Attributes
5.3.6 Submodel Element Attributes
5.3.7 Overview of Submodel Element Types
5.3.8 Concept Description Attributes
5.3.9 Environment Attributes
5.3.10 Referencing in Asset Administration Shells
5.3.11 Primitive and Simple Data Types
5.3.12 Constraints: Global Invariants
6 Data Specification Templates (normative)
6.1 Introduction
6.1.1 Data Specification Template Attributes

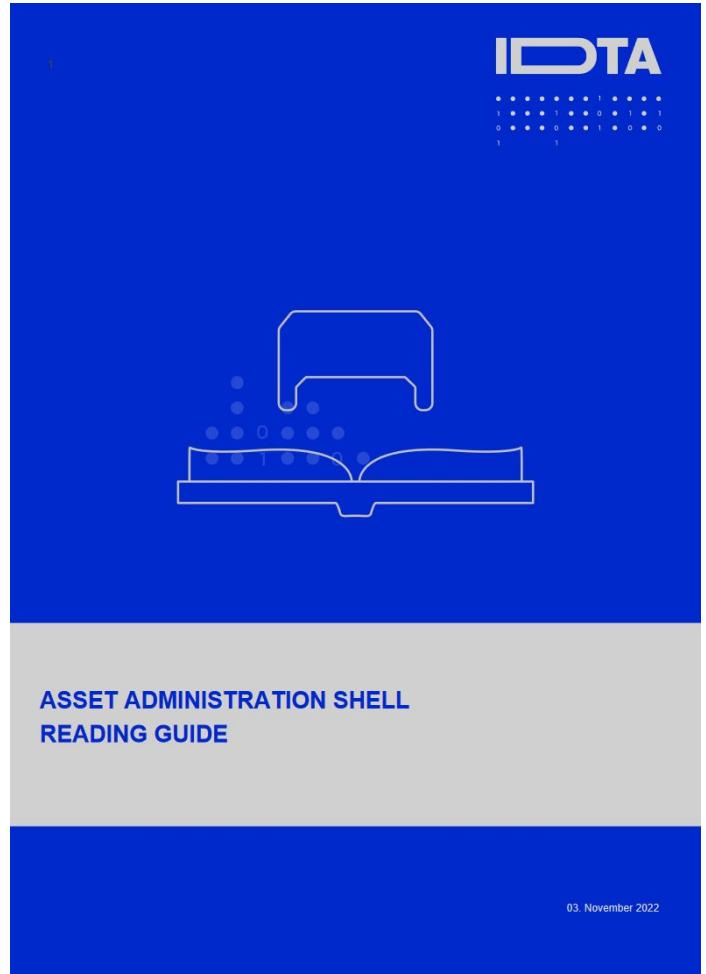
For whom is this tutorial?

- You feel disappointed and do not know how to start?

Have a look at the Asset Administration Shell Reading Guide!

It is updated on a regular basis.

- ▶ **Where to start:** If you have never heard of the AAS
- ▶ **For the generally interested reader:** If you want to learn more about the subject
- ▶ **For decision makers:** If you are interested in the business side of I4.0
- ▶ **For software developers and architects:** If you want to know how to create software for the AAS
- ▶ **For users of the AAS and domain experts:** If you are interested in using the AAS for specific tasks
- ▶ **Security and AI:** If you want to deep dive into these special topics.





● ● ● ● ●
● ● 0 ● 0
● 1 ● 1 ●
● 0 ● ● ●
● 1 ● ● ●

Get started

Download Specification



<https://industrialdigitaltwin.org/content-hub/>

The screenshot shows a document interface with a sidebar on the left containing a 'Bookmarks' list and the main content area on the right.

Bookmarks:

- > 1 Preamble
- > 2 Terms, Definitions and Abbreviations
- > 3 Introduction
- > 4 General
- > 5 The Information Metamodel of the Asset Administration Shell (normative)
- > 6 Data Specification Templates (normative)
- > 7 Mappings to Data Formats to Share I4.0-Compliant Information (normative)
- > 8 Summary and Outlook
- > Annex A. Concepts of the Administration Shell
- > Annex B. Requirements
- > Annex C. Backus-Naur-Form
- > Annex D. Templates for UML Tables
- > Annex E. Legend for UML Modelling
- > Annex F. How to Use the Metamodel
- > Annex G. Metamodel UML with Inherited Attributes
- > Annex H. Metamodel Changes
- > Annex I. Bibliography

Main Content Area:

IDTA April 2023

Specification of the Asset Administration Shell

Part 1: Metamodel

SPECIFICATION

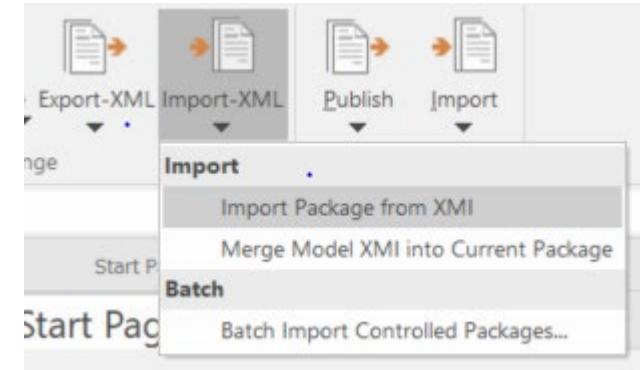
IDTA Number: 01001-3-0

Import XMI to your UML tooling

1.



2.



Releases 7

AAS Schemas V3.0.6 Latest
3 weeks ago

+ 6 releases

<https://github.com/admin-shell-io/aas-specs/releases>

admin-shell-io / aas-specs

Code Issues 8 Pull requests

master aas-specs / schemas / xmi /

<https://github.com/admin-shell-io/aas-specs/tree/master/schemas/xmi>

1. Fetch release of AAS you are interested in
2. Import xmi file into UML tool (best with Enterprise/Architect)

Metamodel Changes

- Annex H. Metamodel Changes
 - General
 - Changes V3.0 vs. V2.0.1
 - Metamodel Changes V3.0 VS. V2.0.1
- Changes V3.0 Vs. V3.0RC02
 - Metamodel Changes V3.0 vs. V3.0RC02

Note for Experts: <Notes for tutorial listeners who have knowledge of previous versions of the specification (V2.0 or Release Candidates of V3.0).

If you do not know previous versions you can ignore these notes.>



14 - Birgit Boss: Details of the Asset Administration Shell V3.0 (Release...

V3.0RC02

<https://www.youtube.com/watch?v=QR-nOl6cuOI>

Annex E. Legend for UML Modelling

OMG UML General

This annex explains the UML elements used in this specification. For more information, please comprehensive literature available for UML. The formal specification can be found in [35].

Figure 63 shows a class with name "Class1" and an attribute with name "attr" of type *Class2*. A owned by the class. Some of these attributes may represents the end of binary associations, s 70. In this case, the instance of *Class2* is navigable via the instance of the owning class *Class*

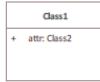


Figure 63 Class

Figure 64 shows that *Class4* inherits all member elements from *Class3*. Or in other word, *Class4* is a generalization of *Class3*, *Class4* is a specialization of *Class3*. This means that each instance of *Class4* is also an instance of *Class3*. An instance of *Class4* has the attributes *attr1* and *attr2*, whereas in *Class3* only have the attribute *attr1*.

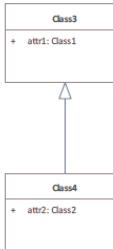
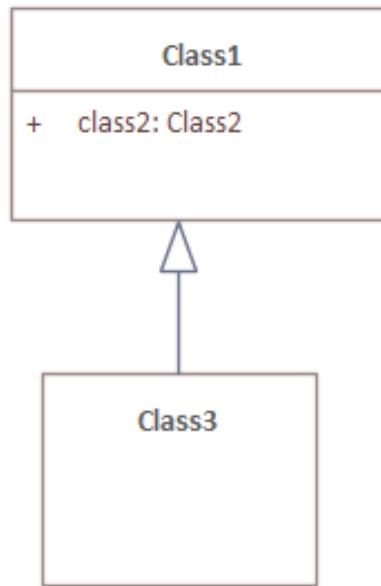


Figure 64 Inheritance/Generalization

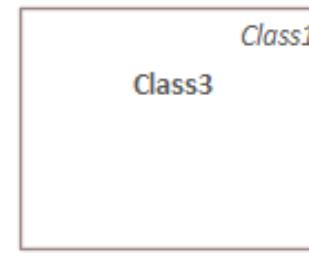
Figure 65 defines the required and allowed multiplicity/cardinality within an association between *Class1* and *Class2*. In this example, an instance of *Class2* is always related to exactly one instance of *Class1*. An instance of *Class1* is either related to none, one, or more (unlimited, i.e. no constraint).

UML Generalization/Inheritance

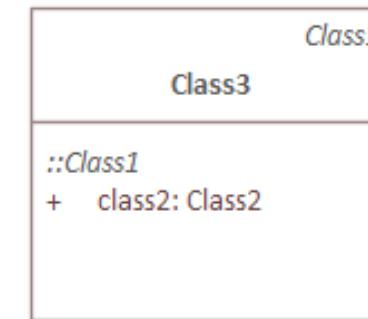
A)



B)



C)

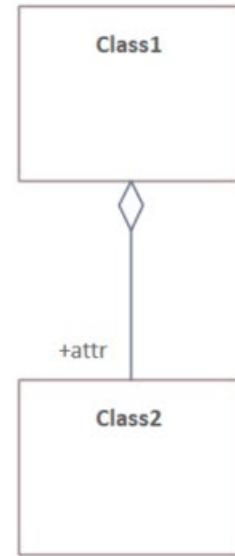


Hint: Graphical representation tool specific

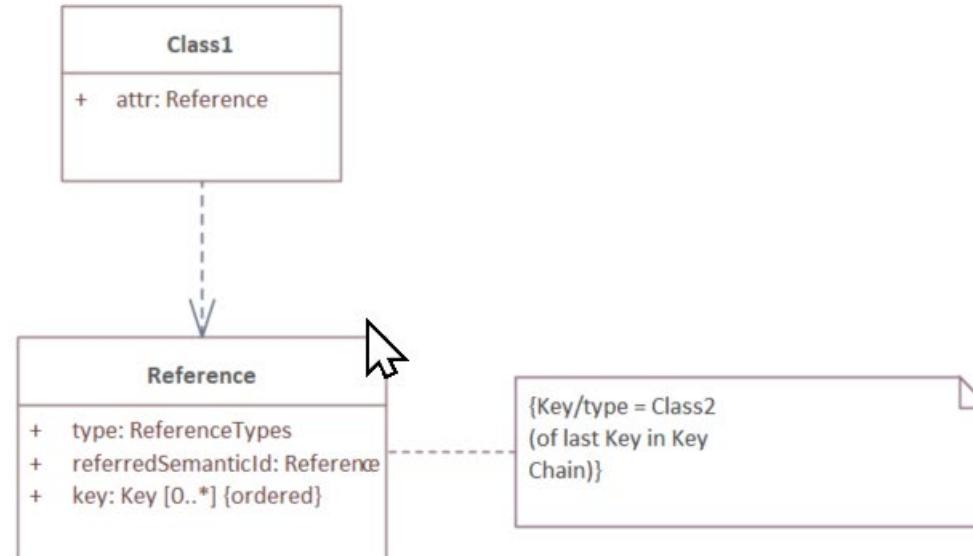
UML Shared Aggregation

Note: Referencing of Referables is an important concept to understand when implementing the AAS

A)

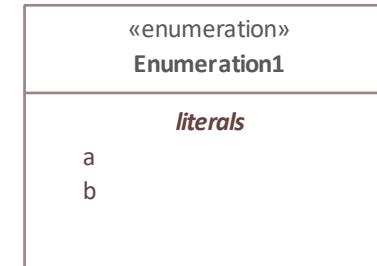
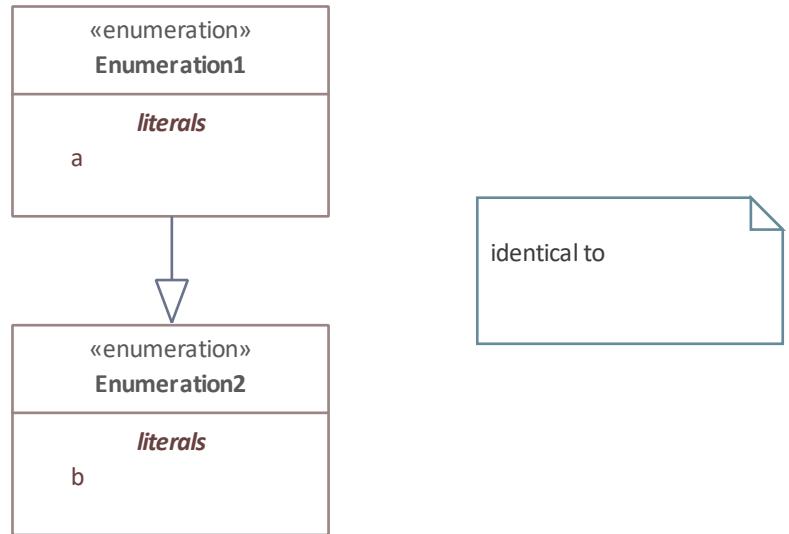


B)



Note for experts: In previous versions a notation of class attributes with reference (*) was used additionally to the notation with the association with the diamond.

Enumerations



Note 1: Inheritance between enumerations is not widely used. It is only used for graphical illustration of relationships between enumerations

Templates



Template for Classes

Template for Classes:

Class:	<Class Name> [<<abstract>>] ["<<Experimental>>"] ["<<Deprecated>>"] [<<Template>>]		
Explanation:	<Explanatory text>		
Inherits from:	{<Class Name> ";" }+ "-"		
Attribute	Explanation	Type	Card.
<attribute or association name> ["<<ordered>>"] ["<<Experimental>>"] ["<<Deprecated>>"]	<Explanatory text>	<Type>	<Card>

Annex D. Templates for UML Tables

- General
- Template for Classes
- Template for Enumerations
- Template for Primitives
- Handling of Constraints

Example for Class Specification

5.3.7.12 Property Attributes

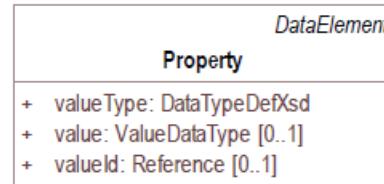


Figure 39 Metamodel of Properties

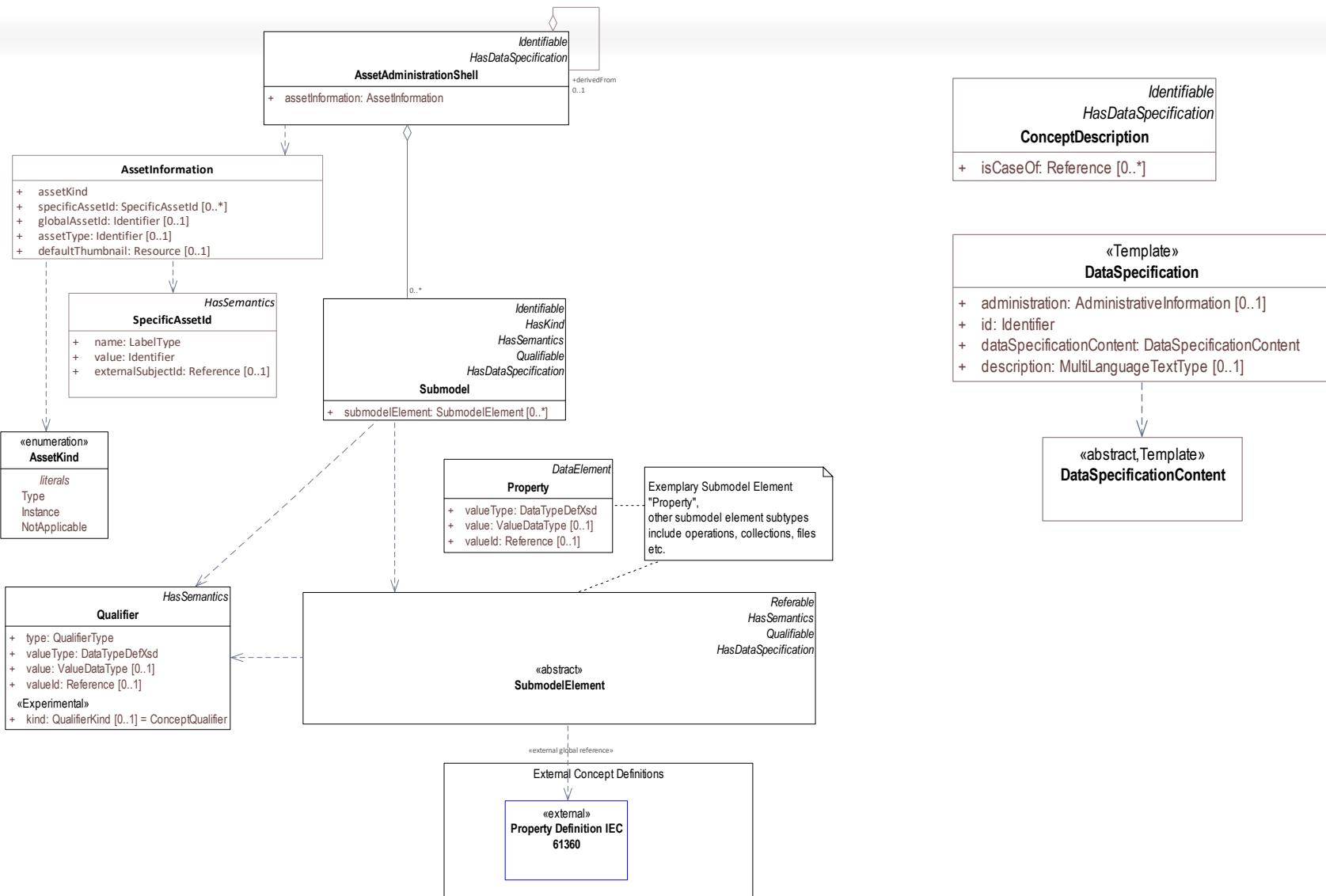
Class:	Property		
Explanation:	A property is a data element that has a single value. <u>Constraint AASd-007:</u> If both the <i>Property/value</i> and the <i>Property/valueld</i> are present, the value of <i>Property/value</i> needs to be identical to the value of the referenced coded value in <i>Property/valueld</i> .		
Inherits from:	DataElement		
Attribute	Explanation	Type	Card.
valueType	Data type of the value attribute	DataTypeDefXsd	1
value	The value of the property instance	ValueDataType	0..1
valueld	Reference to the global unique ID of a coded value Note: it is recommended to use an external reference.	Reference	0..1



0 0
0 0
1 1
0 0
1 1

Get warm

Overview – the goal is to be able to read this diagram



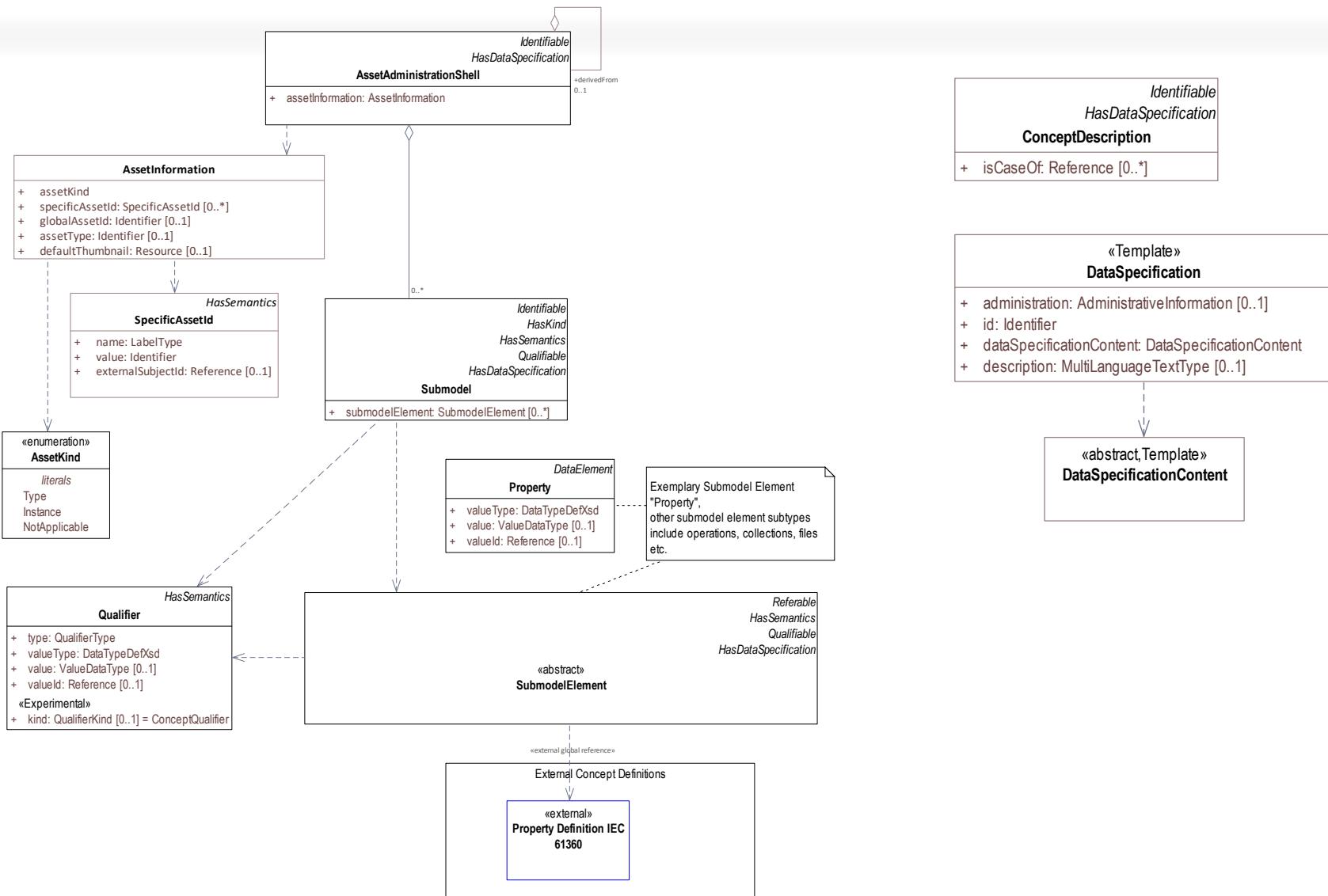
Overview – the goal is to be able to read this diagram



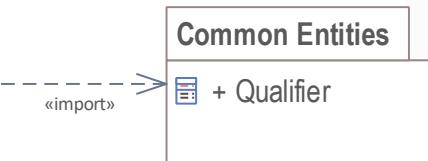
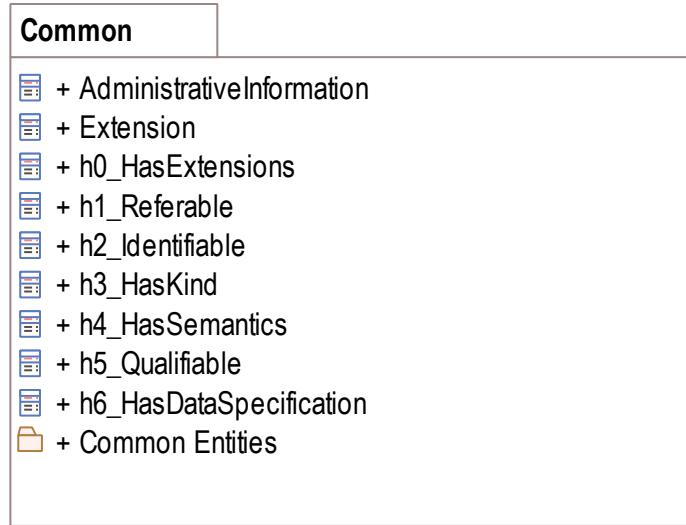
Note for Experts: no views supported any longer

Note for Experts: no assets supported any longer,
AssetAdministrationShell /**assetInformation** introduced instead

Note for Experts: Security in Part 4



Common

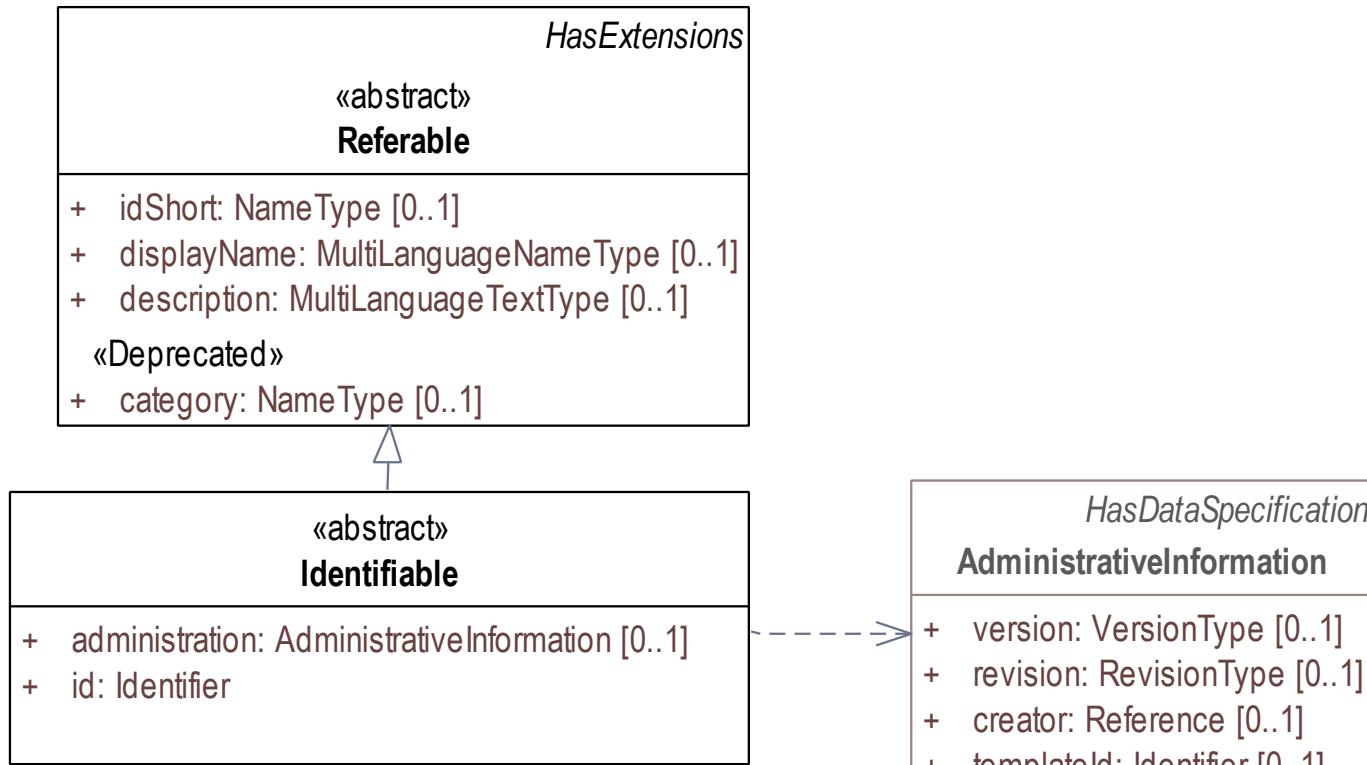


«import»

- 5.3.2 Common Attributes
 - 5.3.2.1 General
 - 5.3.2.2 Administrative Information Attributes
 - 5.3.2.3 Has Data Specification Attributes
 - 5.3.2.4 Extensions Attributes
 - 5.3.2.5 Has Kind Attributes
 - 5.3.2.6 Has Semantics Attributes
 - 5.3.2.7 Identifiable Attributes
 - 5.3.2.8 Qualifiable Attributes
 - 5.3.2.9 Qualifier Attributes
 - 5.3.2.10 Referable Attributes

Note: h0_ h1_ are just added for sorting. In diagrams alias are used without this prefix. Only in package overview and inheritance alias are not supported by the UML tooling used.

Common – Identifiables and Referables



Note for Experts:
Identifier in previous versions of the specification had two attributes: the ID itself and the ID type (IRI, IRDI, Custom). The ID type was removed from the model.

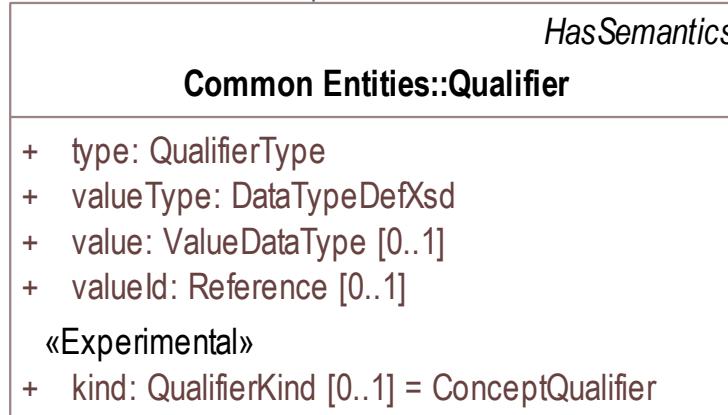
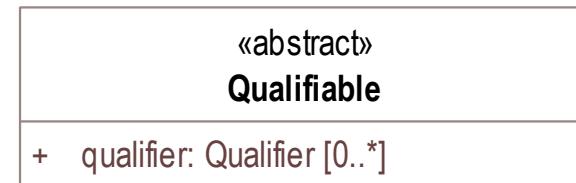
Note for Experts: idShort now optional but still required for non-identifiable referables. DisplayName introduced.

Note for Experts: category of referables set to deprecated.

Note for Experts: New attributes in administrative information: creator and template ID

Note for Experts 3.0: checksum handling in discussion, not part of model 3.0.

Common - Qualifiables



IEC International Electrotechnical Commission
IEC 61360-4 - Common Data Dictionary (CDD - V2.0014.0017)

Search: Qualifier OK

In: Classes Properties
 Value lists Value terms
 Units Lists of Units
 Relations DET classification
 All kind of items

Higher level classes:

Classifying DET:

Properties:

0112/2///61360_4#AAF581 - applicability qualifier
0112/2///61360_4#AAF582 - value origin qualifier
0112/2///61360_4#AAF583 - value processing qualifier
0112/2///61360_4#AAF575 - life cycle qualifier
0112/2///61360_4#ADA356 - operational state qualifier
....

hit Export selected | Select all | Deselect all
 0112/2///61360_4#AAB001 list of qualifiers

Properties tree:

0112/2///61360_4#AAB001 - list of qualifiers
0112/2///61360_4#AAF581 - applicability qualifier
0112/2///61360_4#AAF582 - value origin qualifier
....

Note for Experts: No
Formulas (or other
Constraints) supported
for Qualifiables any
longer,

Note for Experts:
Experimental Qualifier
kind introduced

Common - HasSemantics

«abstract» HasSemantics

- + semanticId: Reference [0..1]
- + supplementalSemanticId: Reference [0..*]

The semanticId is the identifier of the semantic definition of the element.

Supplemental semantic IDs can be added.

Note for Experts: Besides the semanticId supplemental semantic IDs are now possible to be added.

Note for Experts: semantic ID now optional but recommended



Matching Strategies for semantic IDs

Note: isCaseOf of concept descriptions and supplemental semantic IDs are not yet considered in the defined matching strategies.

Exact Matching (identical semantic IDs) – DEFAULT

0173-1#02-AAO677#002

ManufacturerName
semanticId =
0173-1#02-AAO677#002

Herstellername
semanticId =
0173-1#02-AAO677#002

Intelligent Matching (compatible semantic IDs)

Ignore Versioning

0173-1#02-AAO677#002

ManufacturerName
semanticId =
0173-1#02-AAO677#002

Herstellername
semanticId =
0173-1#02-AAO677#003

Consider Semantic Mappings

nominalCapacity
0173-1#02-AAI048#00

nominalCapacity
semanticId =
0112/2///61360_4#AAE530

nominalCapacity
semanticId =
0173-1#02-AAI048#00

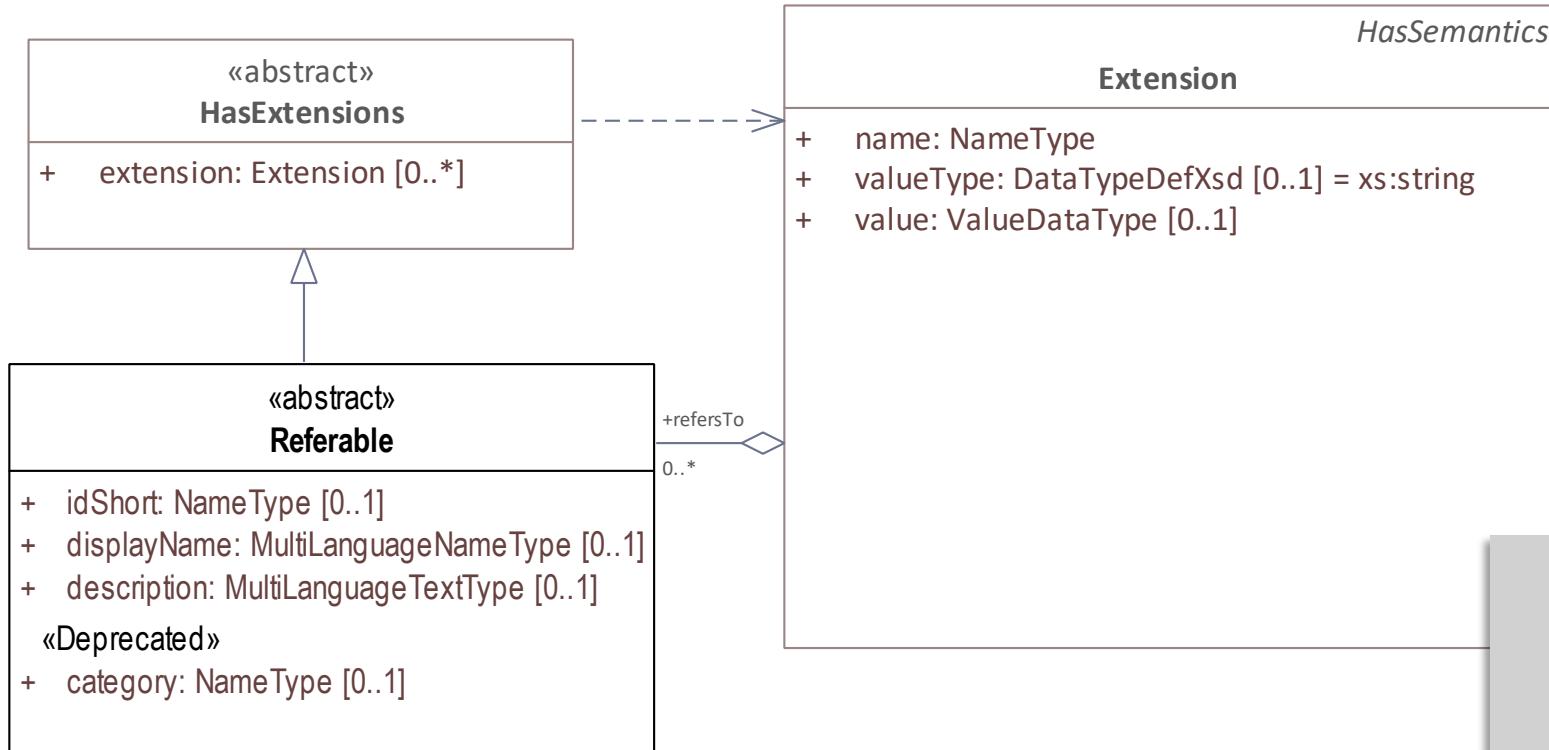
Consider Domain Knowledge

Drill for mineral material 0173-1#01-ADN177#005

HammerDrill
semanticId =
0173-1#01-ADS698#010

PercussionDrill
semanticId =
0173-1#01-ADN177#005

Common - HasExtensions



Note for Experts 3.0:
refersTo now only
supports references to
Referables, not to
external sources.

Note for Experts 2.0:
proprietary extensions
now supported

Allows to annotate an object with
proprietary add-ons (extensions)
without need to (wait for) update the
metamodel

-
Be aware: extensions do not support
interoperability!

Common - HasDataSpecification

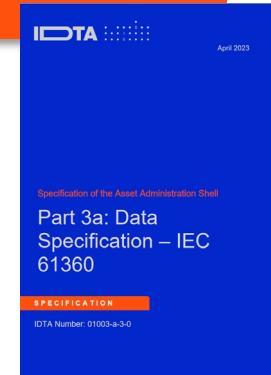


Allows to define
standardized templates for
data specification

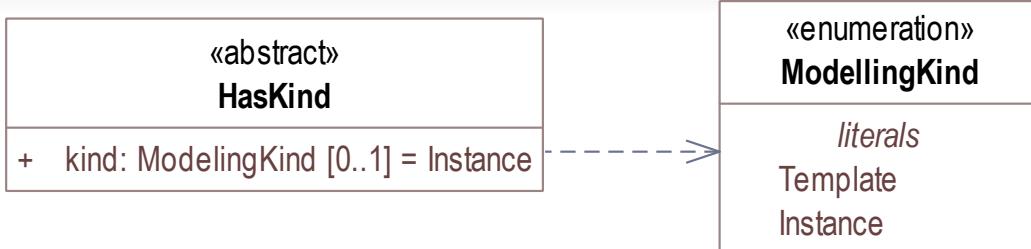
Data specification reference
shall be globally unique and
identifies which data
specifications are used for
an object

Attributes defined in
template are added to the
object

Note for Experts: Data
Specifications are not part of
Part 1 any longer: They are
part of data specifications
series Part 3



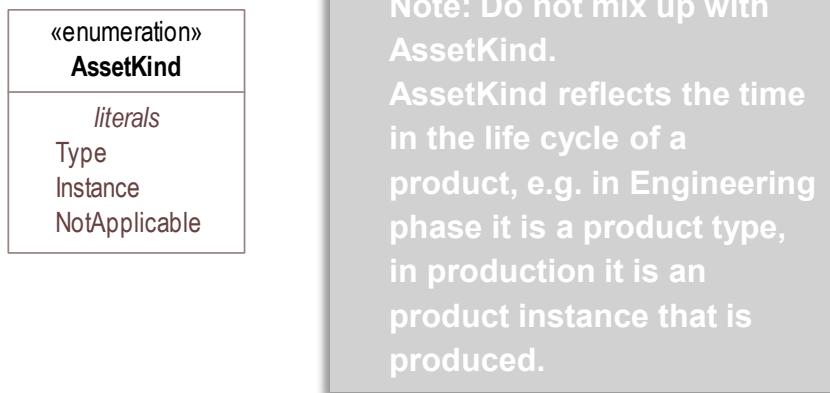
Common - HasKind



ModelingKind = Instance provides values for all data elements within a submodel

ModelingKind = Template is used to specify Submodel Templates, e.g. for the Digital Nameplate or Technical Data

Note for Experts: Only Submodels have the kind attribute, submodel elements do implicitly have the same modelling kind.



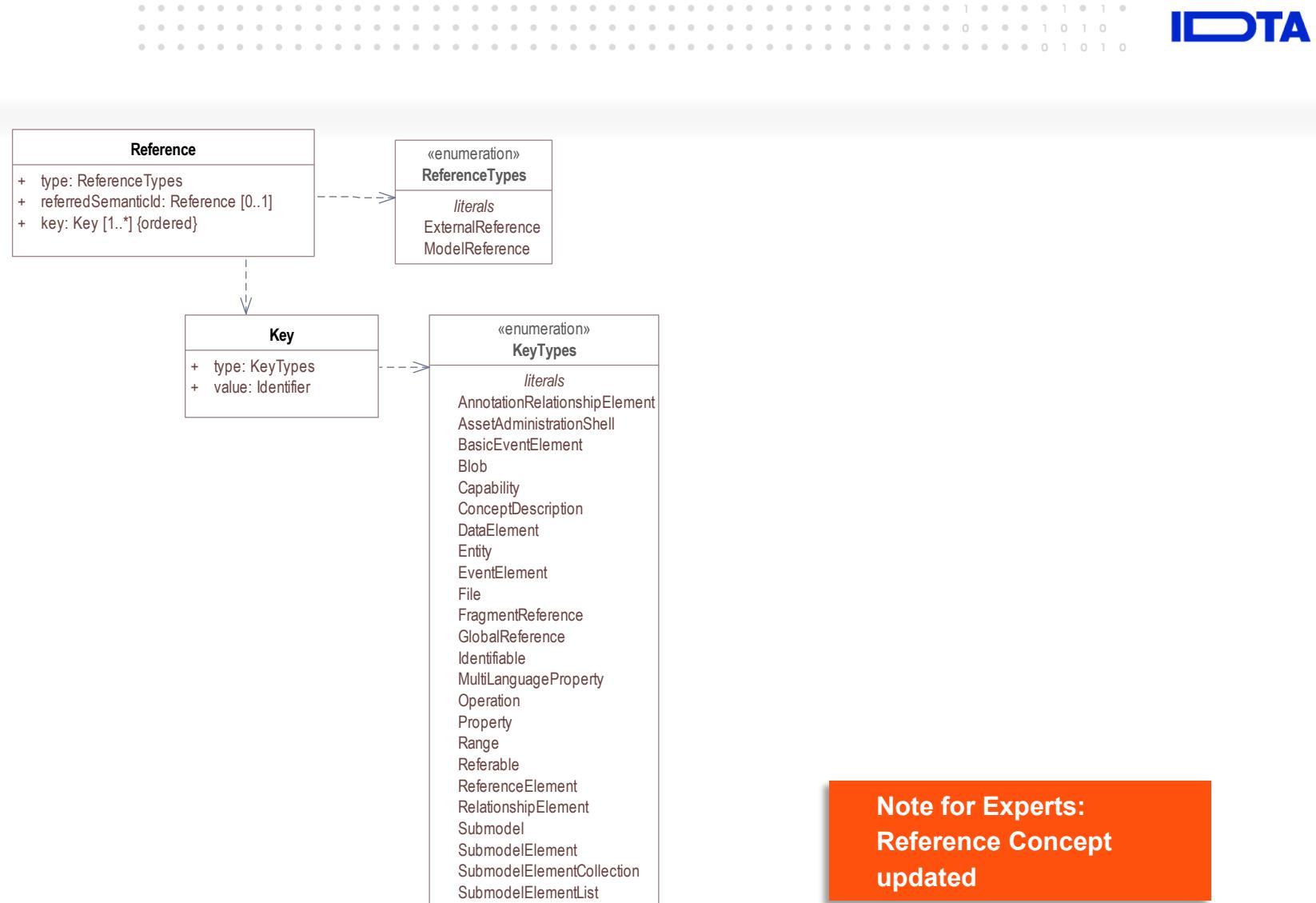
The screenshot shows the IDTA Content Hub interface. The top navigation bar includes 'Home', 'About IDTA', 'Use Cases', 'Technology', 'Content Hub' (which is highlighted), and 'News & Dates'. A search bar is also present. The main content area is titled 'AAS Submodel Templates' and contains a grid of submodel templates. Below the grid, there is a section titled 'Registered AAS Submodel Templates' with a table listing submodel templates. At the bottom, there is a GitHub repository link: 'admin-shell-io / submodel-templates Public'.

Referencing

External
Global
References (e.g.
to ECLASS IRDI,
manfuacter Web-Site)

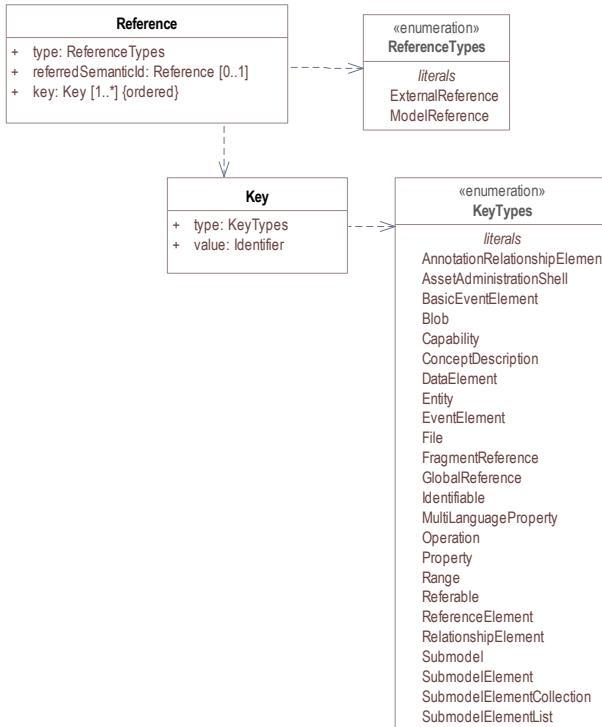
References
into a File
(Fragment)

Model
References to
any Referable
in an AAS
model (e.g. to define
relationships between
elements)



Note for Experts:
Reference Concept
updated

Referencing - Examples



7.2.3 Serialization of Values of Type "Reference"

Some mappings or serializations convert the type "Reference" into a single string. In this case, the following serialization is required:

Grammar:

```

<Reference> ::= "[" <ReferenceType> [ "-" <referredSemanticId> " -" ] "]" ]
<Key> { (", " <Key> )* }

<ReferenceType> ::= "ExternalRef" | "ModelRef"      value of AAS:Reference/type

<SemanticId> ::= "[" <ReferenceType> "]" <Key> { (", " <Key> )*
                                         value of AAS:Reference/referredSemanticId

<Key> ::= "(" <KeyType> ")" <KeyValue>

<KeyType> ::= value of AAS:Key/type

<KeyValue> ::= value of AAS:Key/value

```

Note 1: an IRI may also contain special symbols like "(", "," and "|". A blank is added before the new key or value to distinguish beginning and end of a new key.

Note 2: *ReferenceType* is optional. It is clear from the first key in the key chain whether the reference is a global or a model reference. The examples in this document therefore do not use this prefix.

Valid Examples:

References:

```

(GlobalReference)0173-1#02-BAA120#008
[ExternalRef](GlobalReference)0173-1#02-BAA120#008
(Submodel)https://example.com/aas/1/1/1234859590, (SubmodelElementList)Documents,
(SubmodelElementCollection)0, (MultiLanguageProperty)Title

```

Model References:

```

(ConceptDescription)0173-1#02-BAA120#008
[ModelRef](ConceptDescription)0173-1#02-BAA120#008
(Submodel)https://example.com/aas/1/1/1234859590, (Property)Temperature
[ModelRef- (ConceptDescription)0173-1#02-BAA120#008
-](Submodel)https://example.com/aas/1/1/1234859590

```

Administration shell

**Serial number
Inverter current**

i700E70

231231 Serial number
0.02 Inverter current

• • • • •
• • 0 • • 0
• • 1 • • 1 •
• • 0 • • • •
• • 1 • • • •

Now dive in

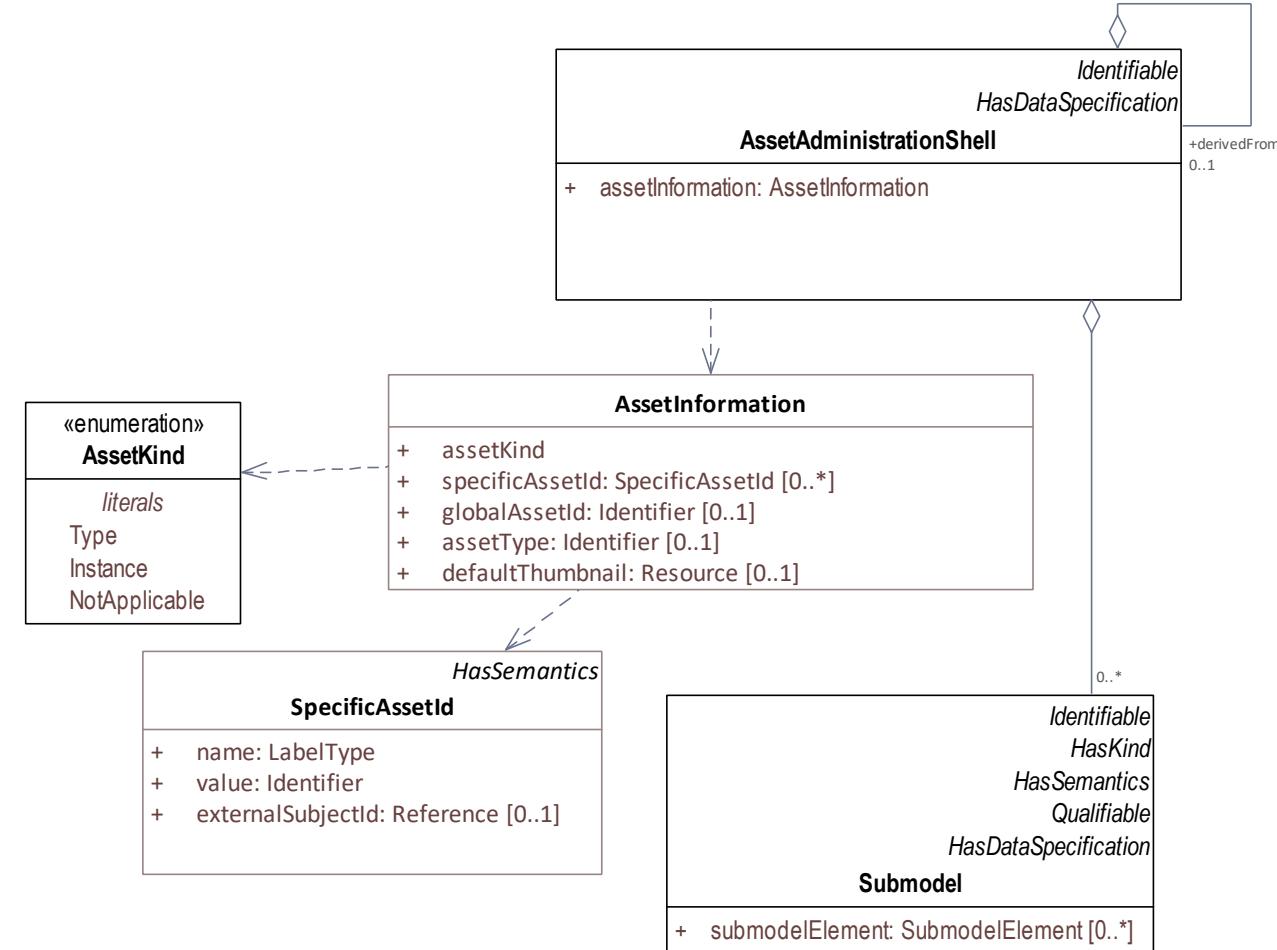
The Asset Administration Shell

Note for Experts: Security
and Asset Administration
Shell now loosely
coupled only

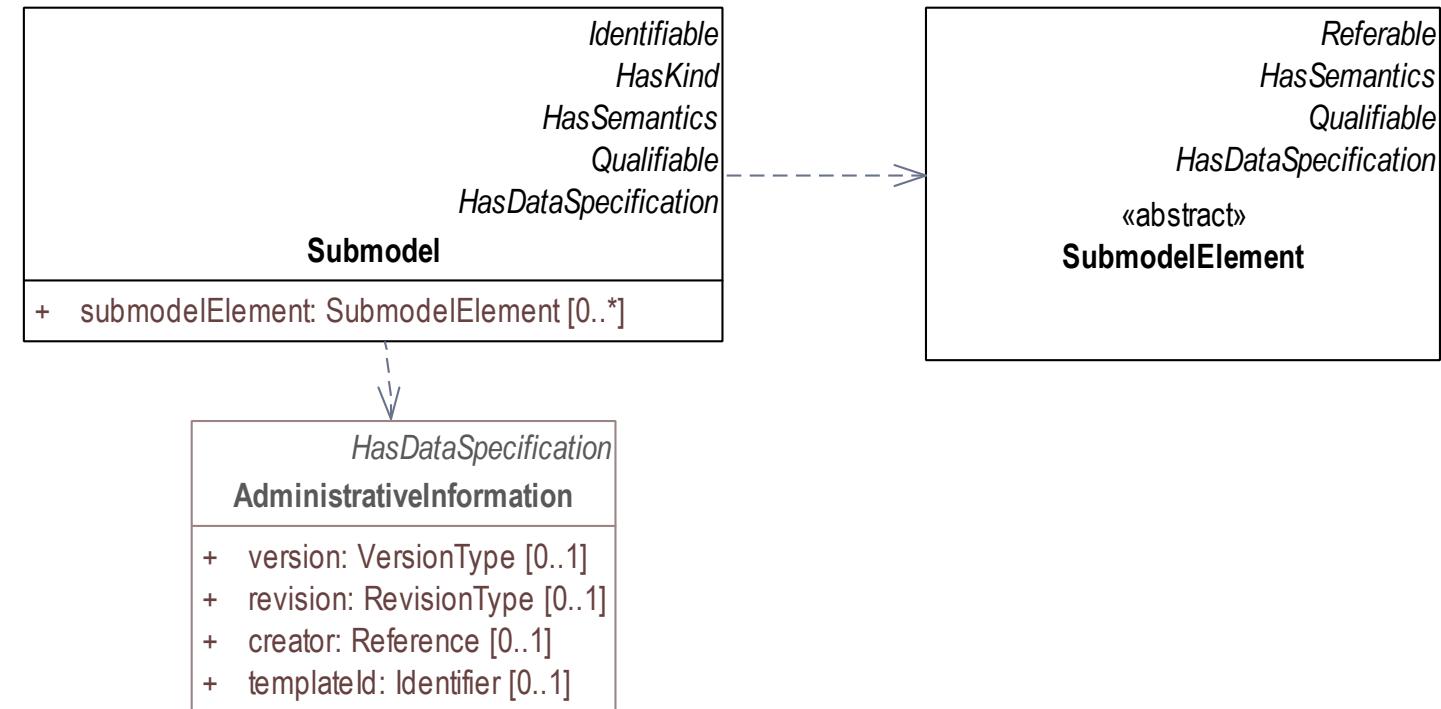
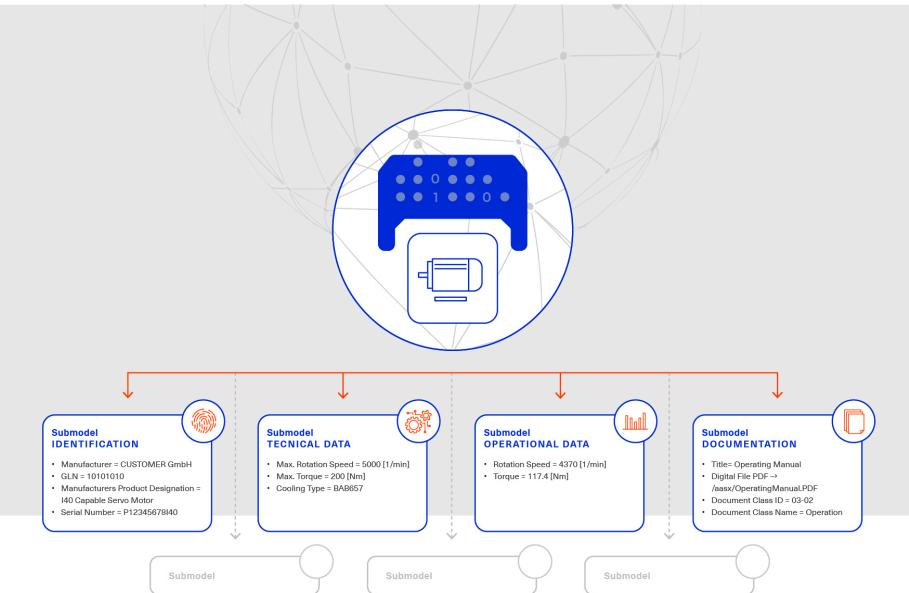
Note for Experts:
Updated asset handling

Note for Experts: new
attribute „assetType“ for
product type – product
instance relationship

Note for Experts: New
value NotApplicable for
asset kind



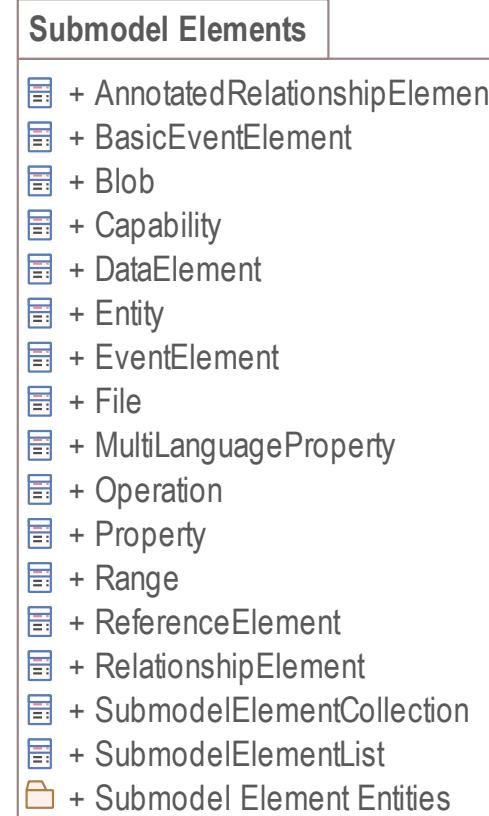
Submodel



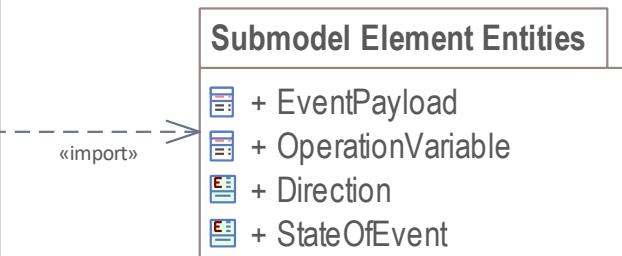
Overview – Package Submodel Elements

5.3.7 Overview of Submodel Element Types

- 5.3.7.1 General
- 5.3.7.2 Annotated Relationship Element Attributes
- 5.3.7.3 Basic Event Element Attributes
- 5.3.7.4 Blob Attributes
- 5.3.7.5 Capability Attributes
- 5.3.7.6 Data Element and Overview of Data Element Types
- 5.3.7.7 Entity Attributes
- 5.3.7.8 Event Attributes
- 5.3.7.9 File Attributes
- 5.3.7.10 Multi Language Property Attributes
- 5.3.7.11 Operation Attributes
- 5.3.7.12 Property Attributes
- 5.3.7.13 Range Attributes
- 5.3.7.14 Reference Element Attributes
- 5.3.7.15 Relationship Element Attributes
- 5.3.7.16 Submodel Element Collection Attributes
- 5.3.7.17 Submodel Element List Attributes

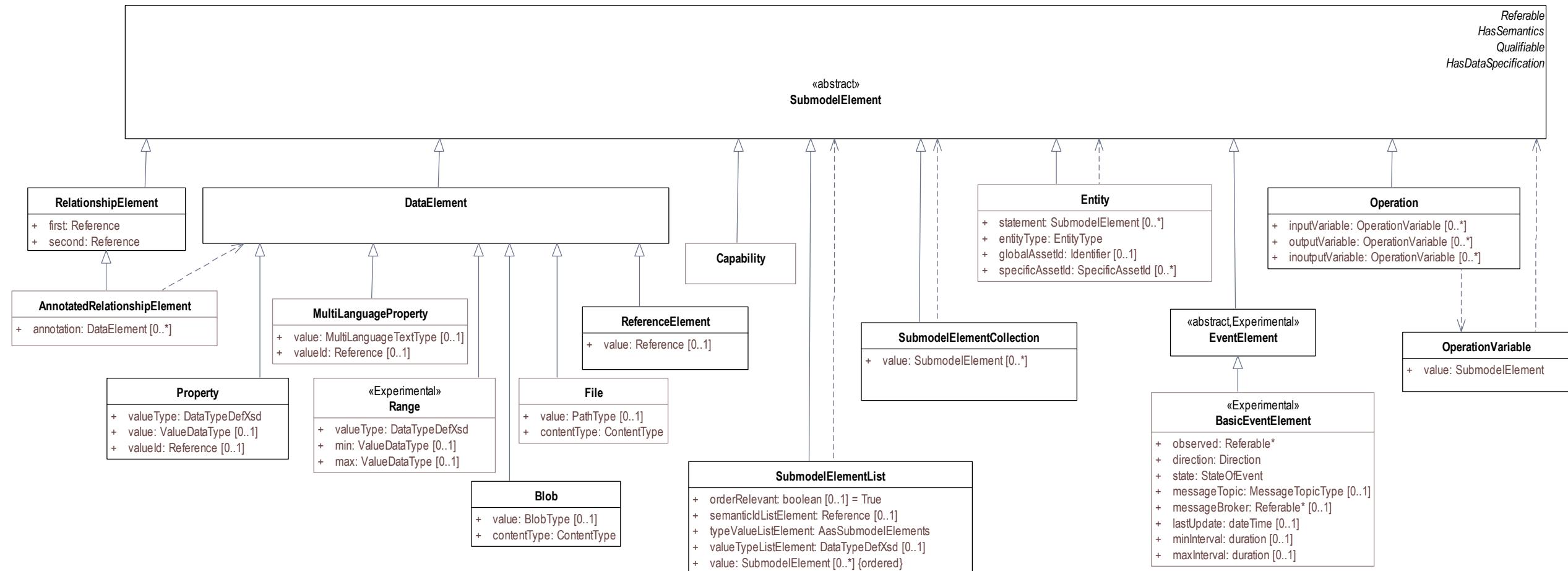


Note for Experts:
SubmodelElementCollection splitted
into **SubmodelElementCollection** and
SubmodelElementList

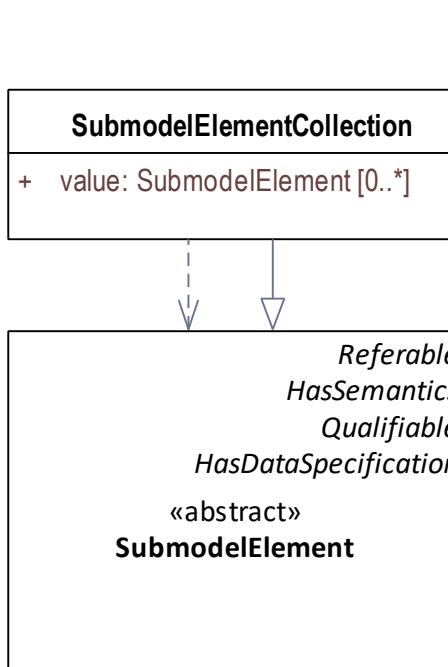


Note for Experts:
experimental
new/updated elements for
events

Submodel Element Subtypes



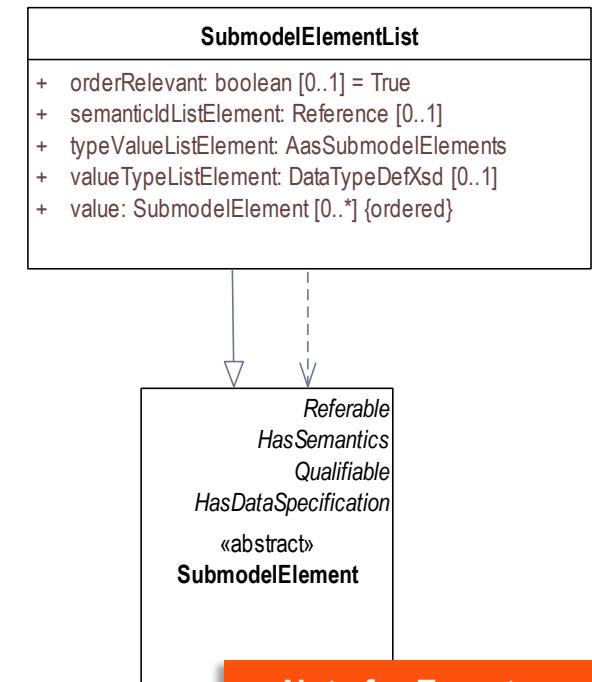
Submodel Element Collections and Lists



```
{
    "NamesOfFamilyMembers": {
        "NameOfMother": "Martha ExampleFamily",
        "NameOfFather": "Jonathan ExampleFamily",
        "NameOfSon": "Clark ExampleFamily"
    }
}
```

Difference in serialization in ValueOnly Format

```
{
    "NamesOfFamilyMembers": [
        "Martha ExampleFamily",
        "Jonathan ExampleFamily",
        "Clark ExampleFamily"
    ]
}
```

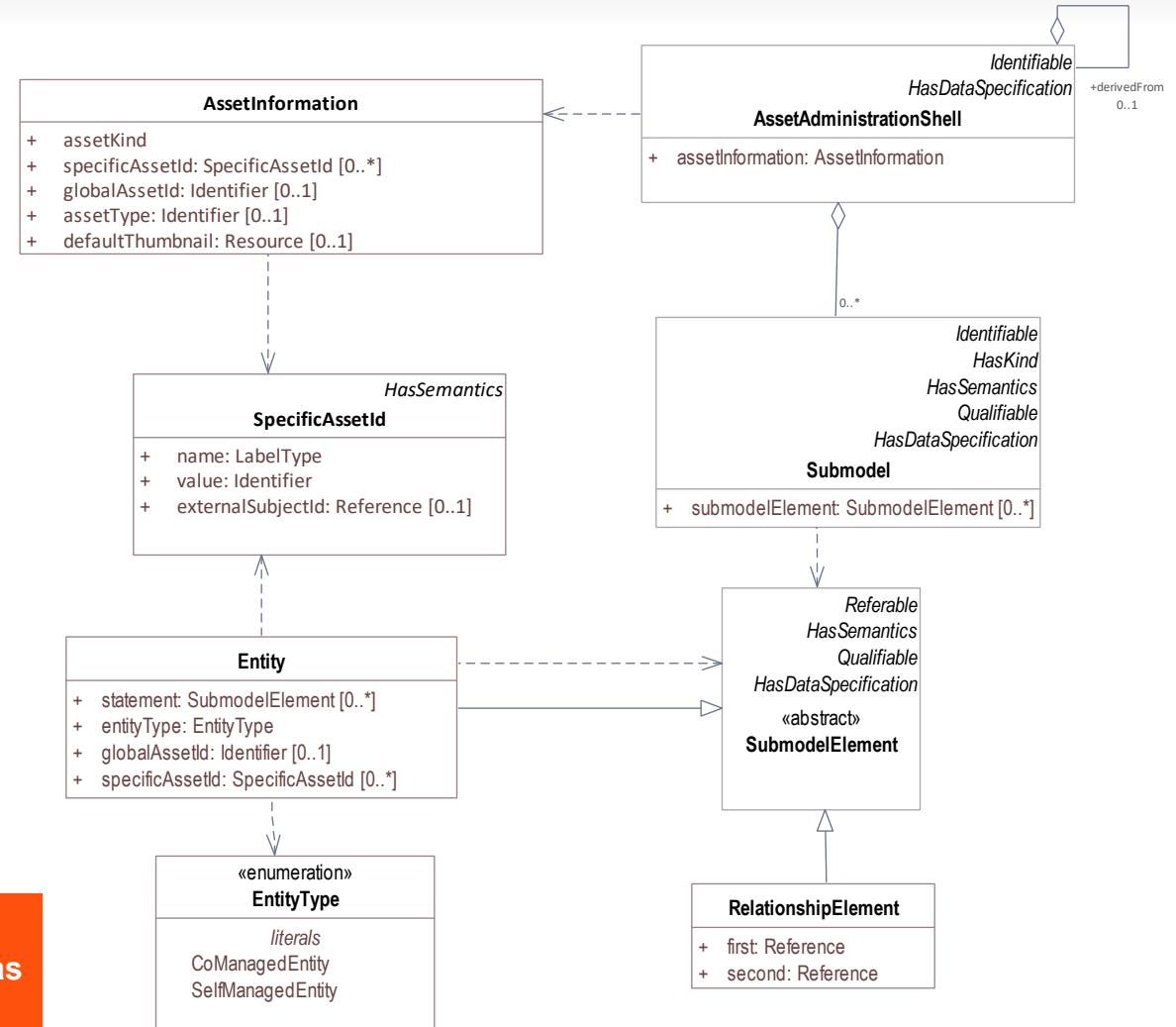


Note for Experts:
Elements in submodel elements list do not have an idShort but are numbered

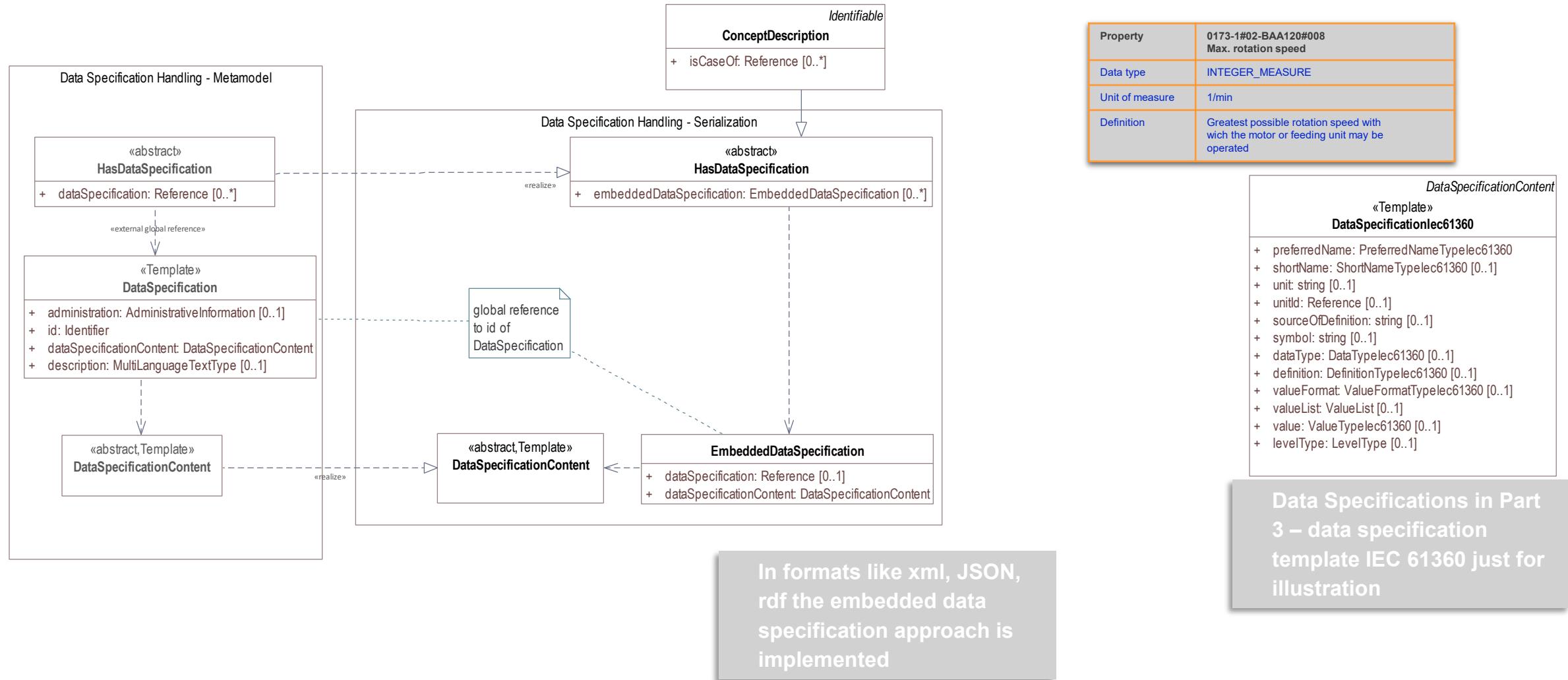
Composite I4.0 Components

- There is no composite Asset Administration Shell, only a composite asset
- Add a bill of material submodel to the AAS of the composite asset: this submodel contains asset IDs to identify its parts
- Via the asset IDs of its parts the AASs of the parts can be found – in case of self-managed entities
- In case of co-managed entity the part is described in the AAS of the composite asset itself
- Any kind of relationship between parts of the composite asset can be expressed

Note for Experts: no bill of materials any longer as part of AssetInformation



Embedded Data Specifications

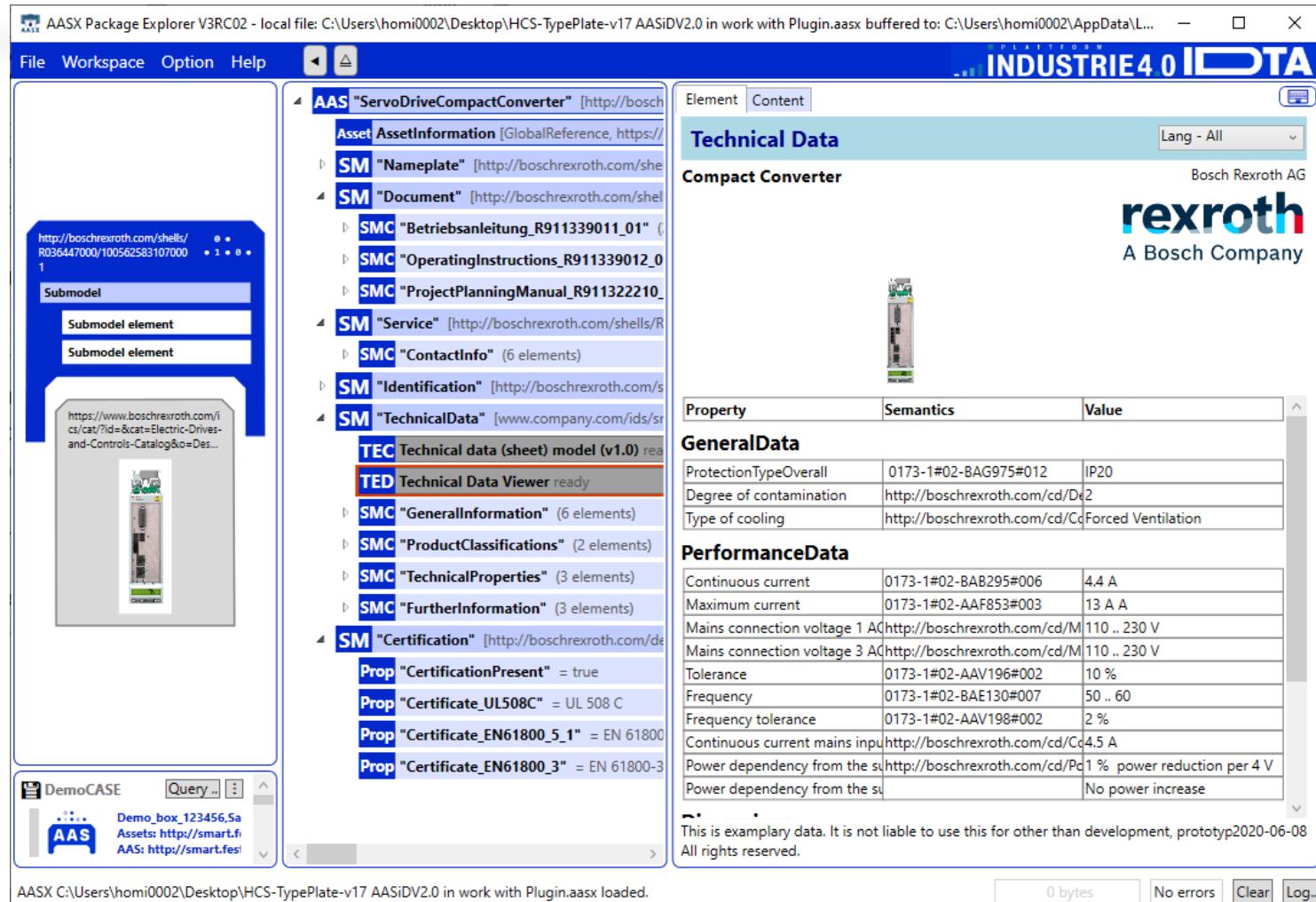




Create your first digital
twin

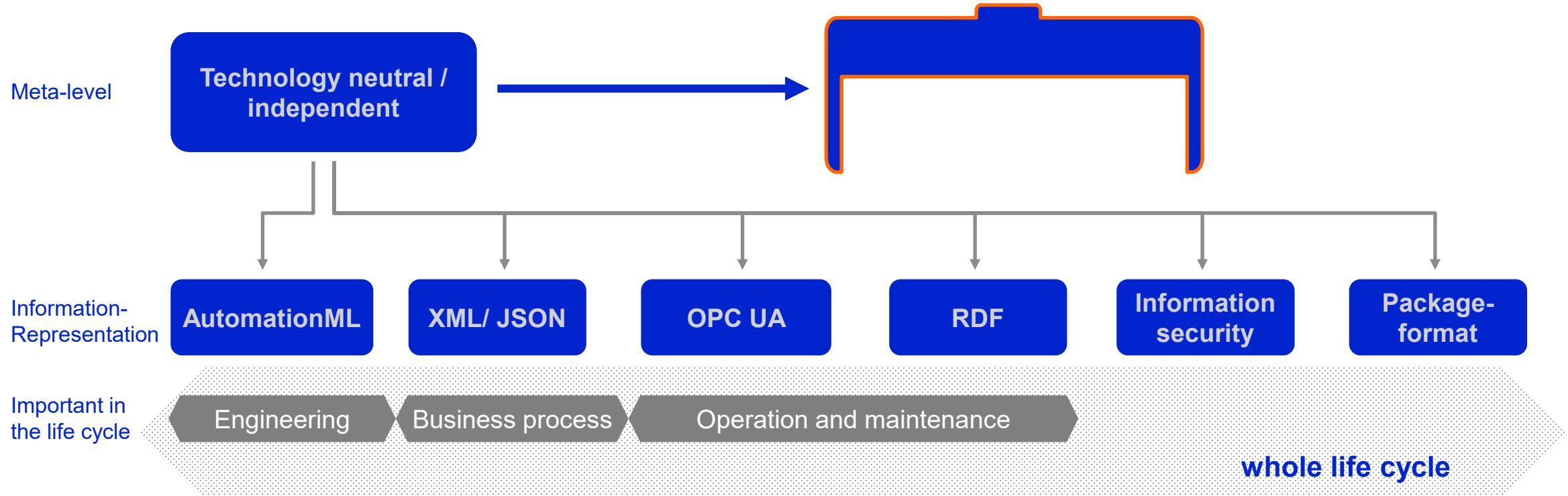
AASX Package Explorer

IDTA



<https://github.com/admin-shell-io/aasx-package-explorer>

Life Cycle Phases and Formats



Serializations/Formats

7 Mappings to Data Formats to Share I4.0-Compliant Information (normative)
7.1 General
7.2 General Rules
7.2.1 Introduction
7.2.2 Encoding
7.2.3 Serialization of Values of Type "Reference"
7.2.4 Semantic Identifiers for Metamodel and Data Specifications
7.2.5 Embedded Data Specifications
7.3 XML
7.4 JSON
7.5 RDF
7.6 AutomationML
7.7 OPC UA

<https://github.com/admin-shell-io/aas-specs/tree/master/schemas>

Note for Experts: Mapping Rules and Schema for xml, JSON and rdf as well as examples not part of specification any longer → now part of open source project admin-shell-io/aas-specs

Note for Experts: Formats like OPC UA or AutomationML are maintained in OPC Foundation and Automation e.V.

README.md

XML

Extensible Markup Language (XML) is a popular serialization format for data exchange and storage.

While there are many possibilities to represent a model of an Asset Administration Shell in XML, we provide our "official" definition (XSD) to foment interoperability between different tools and systems.

Below we explain in more detail how our schema is constructed, point the user to the examples and finally give some basic on our particular schema design.

Top-Level Structure

The root element of our XML is an XML element representing the instance of `Environment`. This environment contains the corresponding to all `Identifiable` classes:

- `AssetAdministrationShell`'s,
- `Submodel`'s, and
- `ConceptDescription`'s.

To simplify exploration of the XML data, identifiable instances are only available at the level of their parent class.

We now continue to see how to serialize the instances and their properties.

Mapping Rules

Building blocks of an XML document include only `XML elements`, `XML attributes` and text enclosed in an element. XML elements can have children elements. Using these building blocks, we map an AAS model to XML.

UML Property to XML Element

Before we look into how to represent instances of classes, let us start bottom-up and see first how individual properties are mapped.

We represent each property of a class with an XML element whose name corresponds to the property name in UML, in camel-case where all abbreviations are left as capitalized (`dataSpecificationIec61360` instead of `dsIec61360`).

It is common in UML to use singular form for aggregations, which is the case for the meta-model. In XML, however, code, where plural form for sequences is common. Since the naming of XML elements has direct influence on the properties in plural form deviating from the name in the meta-model. For example, `submodels` is plural, while `submodel` is singular.

	json
	rdf
	xmi
	xml
	yaml
	.gitignore
	InstallSchemaValidation.ps1
	Validate.ps1

Note: for data specifications the embedded approach is used

Note: see Readme files for different mappings to XML, JSON and RDF

Open Source Support



Search or jump to... / Pull Requests

admin-shell-io by IDTA
Industrial Digital Twin Association e.V.
<https://idtwin.org/>

[https://github.com/orgs
/admin-shell-io/](https://github.com/orgs/admin-shell-io/)

Note: specifications
maintained in admin-shell-io

ECLIPSE FOUNDATION

Projects Working Groups

Home / Projects / Eclipse Digital Twin / Governance

Eclipse Digital Twin

Overview Downloads Who's Involved Developer Resources Governance Contact Us

Scope:
The Eclipse Digital Twin Top-Level Project supports projects at the Eclipse Foundation focusing on the implementation of solutions, prototypes and supporting software of digital twin technology.

The envisioned efforts include the following areas:

- Modelling and building digital twins based on open standards and technologies
- Modelling and consuming of existing and new open standards for the information provided via digital twins (dictionaries and semantic models/ontologies) components and modules for digital twins
- Infrastructural components for developing and operating digital twins
- Graphical User Interfaces for visualizing and interacting with digital twins
- Backend adapters for gathering data provided via digital twins in standardized formats
- Connection of digital twins with existing semantic dictionaries and ontologies
- Usage of digital twins in federated infrastructures
- Support of static (master data), dynamic (runtime) and behavioural data across the complete life cycle of an asset represented by a digital twin
- Lifecycle Management of digital twins
- Support of different development, testing, deployment, and operation strategies of digital twins
- Integration of digital twins with other technologies
- Development examples and demonstrators of digital twins and tools

<https://projects.eclipse.org/projects/dt/>

RELATED PROJECTS ▾

Project Hierarchy:

- » Eclipse Digital Twin
- » Eclipse AAS Model for Java
- » Eclipse AAS Web Client
- » Eclipse AASX Package Explorer
- » Eclipse BaSyx™
- » Eclipse Semantic Modeling Framework
- » Eclipse Service Lifecycle Management

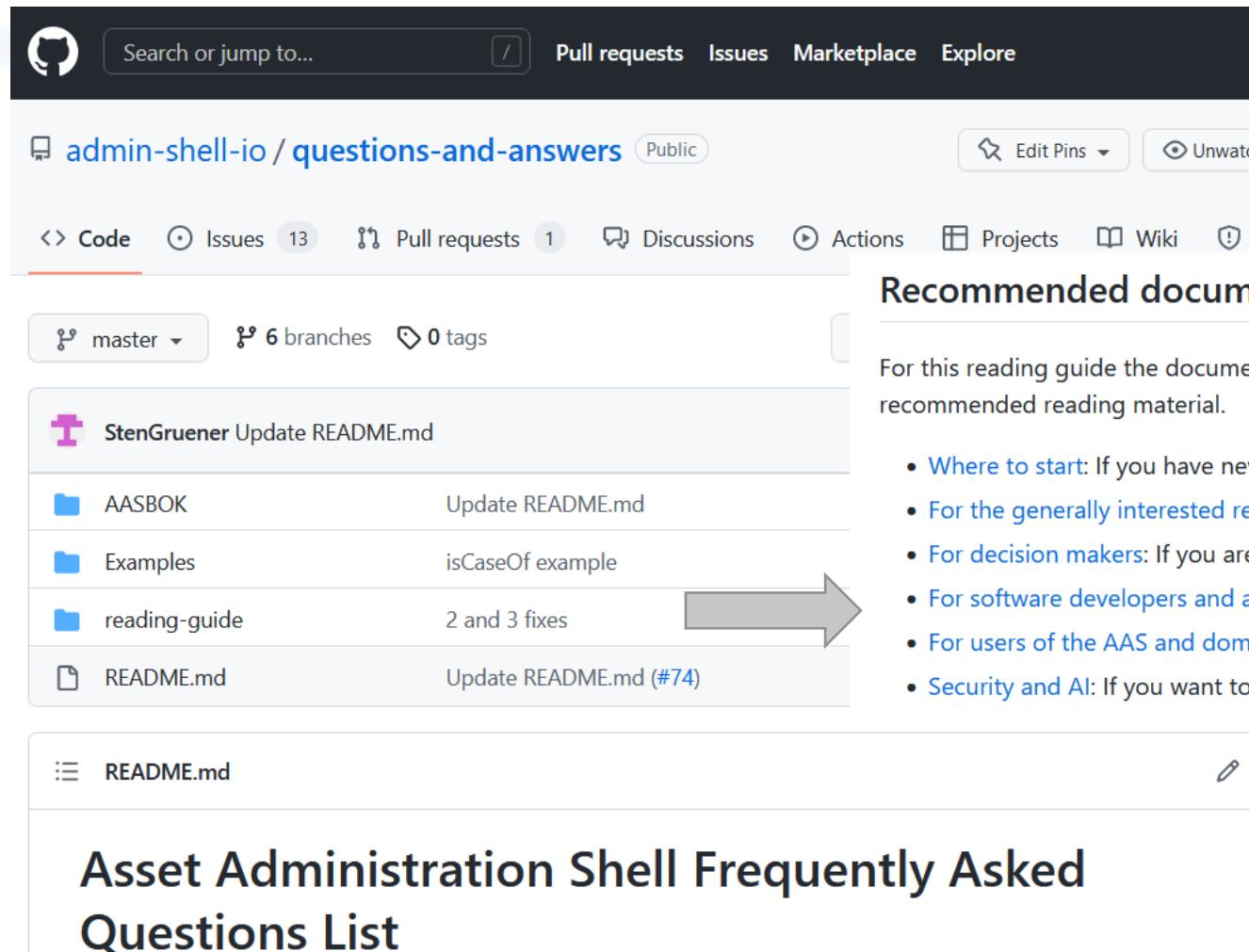
Status: April 2023



•	•	•	•	•	•
•	•	0	•	0	•
•	•	1	•	1	•
•	•	0	•	•	•
•	•	1	•	•	•

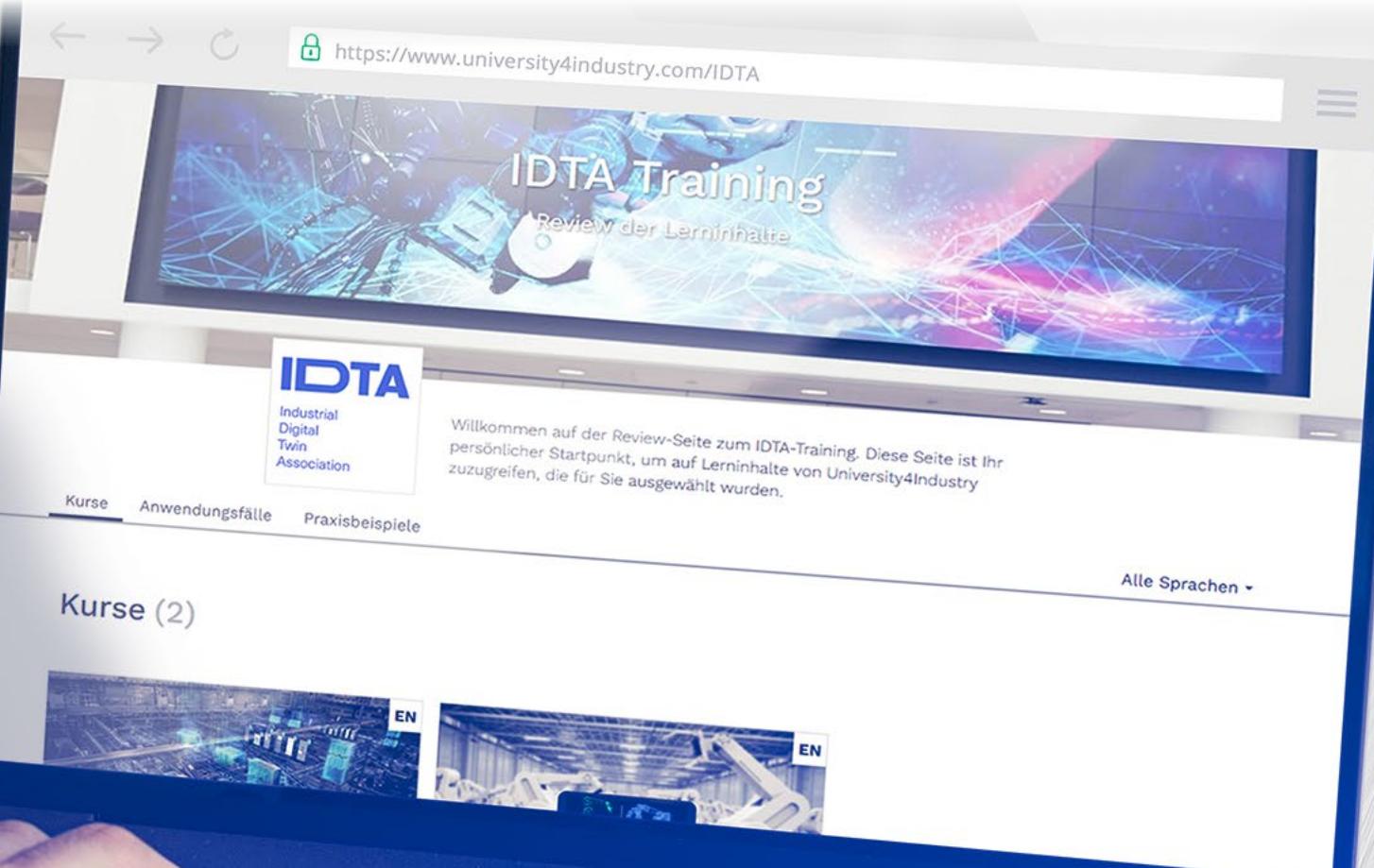
Still Questions?

Questions and Answers



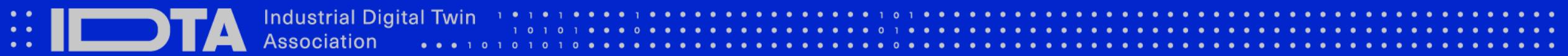
The screenshot shows a GitHub repository page for 'admin-shell-io/questions-and-answers'. The repository is public and has 6 branches and 0 tags. A pull request by StenGruener titled 'Update README.md' is visible. The 'Code' tab is selected. In the top right, there are 'Edit Pins' and 'Unwatch' buttons. Below the navigation bar, there are links for 'Issues' (13), 'Pull requests' (1), 'Discussions', 'Actions', 'Projects', 'Wiki', and a shield icon. The main content area is titled 'Recommended documents' and contains a message about sorting documents by interest groups. It lists several categories: 'Where to start', 'For the generally interested reader', 'For decision makers', 'For software developers and architects', 'For users of the AAS and domain experts', and 'Security and AI'. A large gray arrow points from the 'reading-guide' pull request towards this list.

<https://github.com/admin-shell-io/questions-and-answers>



The screenshot shows a web browser displaying the URL <https://www.university4industry.com/IDTA>. The page title is "IDTA Training" and the subtitle is "Review der Lerninhalte". The background features a blue-toned industrial scene with a robot arm. On the left, there's a sidebar with the IDTA logo and navigation links for "Kurse", "Anwendungsfälle", and "Praxisbeispiele". The main content area has a welcome message: "Willkommen auf der Review-Seite zum IDTA-Training. Diese Seite ist Ihr persönlicher Startpunkt, um auf Lerninhalte von University4Industry zuzugreifen, die für Sie ausgewählt wurden." Below this, there's a section titled "Kurse (2)" with two thumbnail images: one showing a factory interior and another showing a robotic arm. A "Alle Sprachen" dropdown menu is visible on the right.

www.u4i.io/IDTA



Let's go!



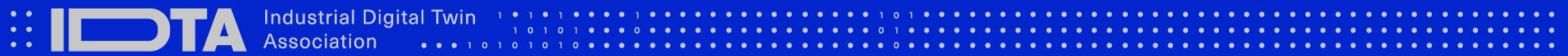
Connect on

www.linkedin.com/in/birgit-boss/

Dr. Birgit Boss

Robert Bosch GmbH, Bosch Connected Industry

- Board member of the Industrial Digital Twin Association (IDTA)
- Chair of the Working Group “Open Technology” and its Working Stream “Specifications of the Asset Administration Shell”
- Chair of the Working Group “Semantic Layer including Digital Twins” of Catena-X
- PMC member of the Eclipse Digital Twin Top Level Project



www.industrialdigitaltwin.org