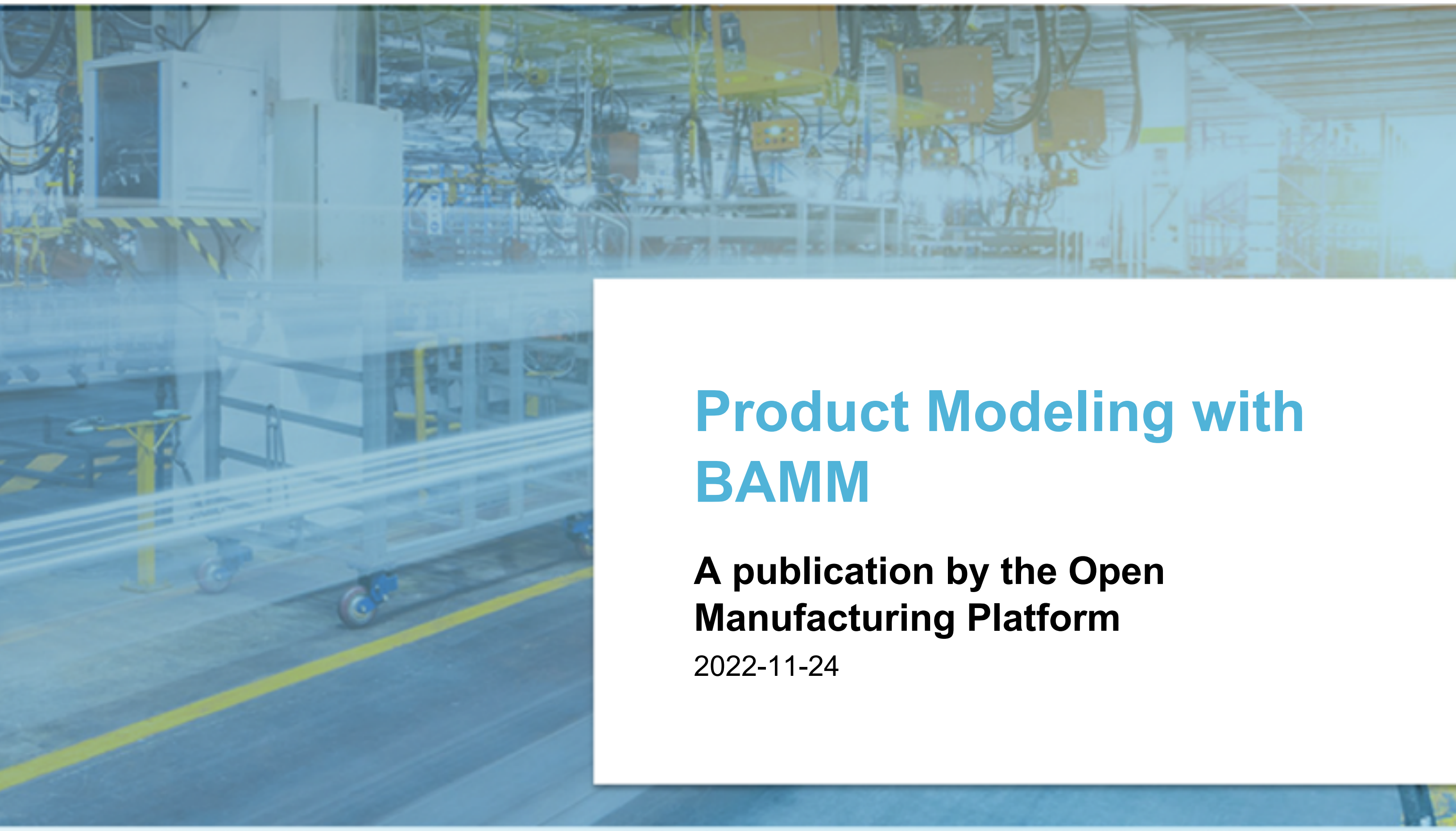
OPEN MANUFACTURING PLATFORM



Product Modeling with BAMM

A publication by the Open Manufacturing Platform

2022-11-24

www.open-manufacturing.org

info@open-manufacturing.org

Legal Disclaimers

ACKNOWLEDGMENTS
This document is a work product of the Open Manufacturing Platform – Semantic Data Structuring Workgroup, chaired by Birgit Boss (Robert Bosch GmbH).

AUTHORS (in alphabetical order)
Andreas Textor (Robert Bosch GmbH)
Anna Vogt (Audi AG)
Birgit Boss (Robert Bosch GmbH)
Chris Volk (Robert Bosch GmbH)
Denis Molin (Teradata Corp.)
Yingbing Hua (SICK AG)

Contents

Table of Figures

1 Introduction

The BAMM Aspect Meta Model has been introduced as a formalization of the specification of semantic models, or aspect models, that describe aspects of a digital twin. This whitepaper shows examples of BAMM for modeling aspects of industrial products' technical data. Figure 1 corresponds to the modeling of device models using the Aspect Meta Model. The data provided by the digital twins again would conform to the device model or – in our use case – a product description.

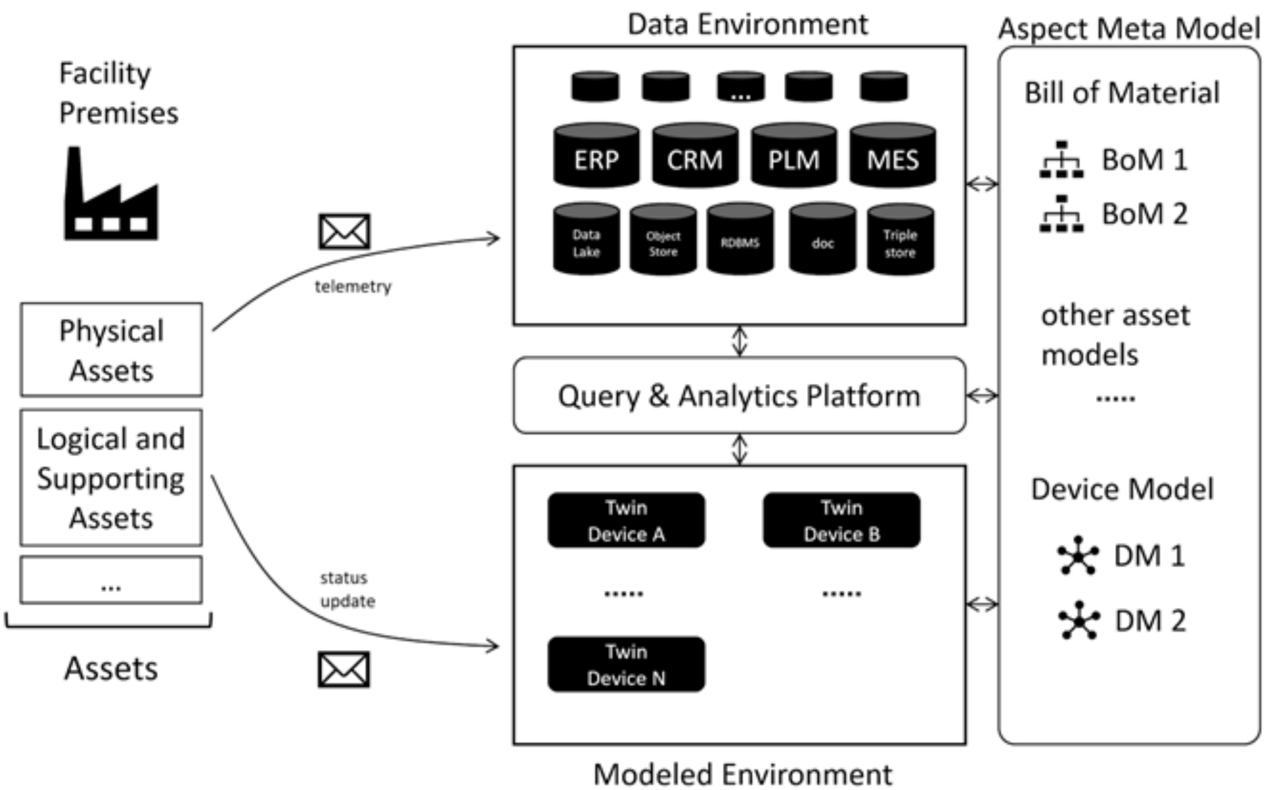


Figure 1 Conceptual Architecture[1]

[1] "Semantic Data Structuring", Open Manufacturing Platform, 2022-08-16. Online: https://github.com/OpenManufacturingPlatform/openmanufacturingplatform.github.io/blob/master/docs/sds/OMP-Semantic-Data-Structuring-Whitepaper.pdf

The modeling of products begins with the modeling of product aspects. Although mature public property dictionaries exist, such as IEC CDD and ECLASS, individual needs can hardly be satisfied based on public repositories alone. Companies often want to define their product characteristics using public standards so that the syntax is interoperable and partners and customers can correctly interpret the semantics.

This paper shows how to define the semantics of a specific aspect of a digital twin, for example, specified by the Asset Administration Shell in Industry 4.0, not using single dictionary entries but by defining the semantics of a complete submodel.

The BAMM Aspect Meta Model allows the definition of properties and characteristics and thus enables the definition of semantics in a very flexible and fast way. The approach dramatically facilitates the reusability of the characteristics of different products, which is vital for cross-domain data integration.

This whitepaper is organized as follows. In Chapter 2, we give a brief introduction to BAMM. Chapter 3 describes the application scenario and the concrete industrial product used as a running example throughout this whitepaper. Chapter 4 demonstrates the utility of BAMM for modeling various kinds of technical data which represent the typical requirements in industrial use cases. Chapter 5 shows how aspect models can be used as a semantic description of submodels of an Asset Administration Shell (AAS). Finally, Chapter 6 summarizes the whitepaper and discusses future work.

2 A brief Introduction to BAMM Aspect Meta Model

An aspect model contains information about runtime data structure (e.g., that there is a property in the data called "temperature" and that it has a numeric value) as well as information that is not part of the runtime data (e.g., the physical unit or value range). It does not, however, contain actual runtime data (e.g., a numeric value representing the current temperature), as this will be delivered by an *Aspect* — a service implementation that provides data — conforming to this aspect model. The combination of raw runtime data and its corresponding aspect model yields information.

An aspect model, therefore, serves two purposes:

1. It captures the domain semantics of the part of the digital twin it describes. In this regard, it can be considered an ontology. Information usually available explicitly but informally (for example, in a datasheet of a machine) or implicitly (as tacit knowledge) is now made explicit and consistent in a broader scope than a single service or application.
2. It serves as a contract between the data producer and data consumer, similar to a schema description (cf., e.g., JSON Schema or XML Schema). In this regard, it predefines which data structures and values may appear in the runtime data. However, unlike a pure schema language, it also contains the previously mentioned domain semantics, which is not included in the runtime data but can partly be used for validation.

3 Application Scenarios

Product modeling, or product type modeling, is essential for various industrial use cases, such as product information management, product catalog management, etc.

In this whitepaper, the modeling capacity of BAMM will be demonstrated using real-world product information from the sensor manufacturer SICK AG. The example being used through this whitepaper is publicly available on the home page of SICK AG[2]. A screenshot of the original webpage is shown in Figure 2.

[2] https://www.sick.com/de/en/photoelectric-sensors/photoelectric-sensors/w16/wtl16p-1h161120a00/p/p512653

Illustration may differ

Type: **WTL16P-1H161120A00** | Part no.: **1218946**

Log in to get your price and the estimated date of arrival

1 Piece **Add to cart** **Request quote**

CE ECOLAB IO-Link PinPoint 2.0 SIRIC

Product data sheet | Copy shortlink | Add to wish list

Technical details | Downloads | Accessories | Videos | Service and Support | Customs data

Show all | Hide all

− Features

Device type	Photoelectric sensors
Functional principle	Photoelectric proximity sensor
Functional principle detail	Background suppression, LineSpot technology
Sensing range	
Sensing range min.	10 mm
Sensing range max.	500 mm
Adjustable switching threshold for background suppression	100 mm ... 500 mm
Reference object	Object with 90 % reflectance (referred to standard white, DIN 5033)
Minimum distance between set sensing range and background (black 6% / white 90%)	15 mm (at a distance of 200 mm)
Recommended sensing range for the best performance	150 mm ... 250 mm

+ Safety-related parameters

+ Communication interface

+ Electrical data

− Mechanical data

Housing	Rectangular
Dimensions (W x H x D)	20 mm x 55.7 mm x 42 mm
Connection	Cable, 4-wire, 2 m
Connection detail	
Deep-freeze property	Do not bend below 0 °C
Conductor size	0.14 mm²
Cable diameter	Ø 4.8 mm
Length of cable (L)	2 m
Bending radius	For flexible use > 12 x cable diameter
Bending cycles	1,000,000
Material	
Housing	Plastic, VISTAL®
Front screen	Plastic, PMMA
Cable	PVC
LABS-free	Yes (VDMA 24364-A1-L)
Weight	Approx. 100 g
Maximum tightening torque of the fixing screws	1.3 Nm

+ Ambient data

+ Smart Task

+ Diagnosis

+ Classifications

Figure 2 Example of product information provided by SICK AG

The product information structure has six major domains (Technical details, Downloads, Accessories, Videos, Service and Support, and Customs data), each of which may contain further sub-domains. For example, technical details constitute Features, Safety-related parameters, Communication interfaces, Electrical data, Mechanical data, Ambient data, Smart Tasks, Diagnosis, and Classifications. Each sub-domain contains a set of attributes that describe the product's properties, such as device type, dimensions, etc.

One key design choice for modeling such a product is to choose the correct level of abstraction. The BAMM meta-model provides the concept **aspect** to encapsulate actual data of the digital twin of a product that clients can use. In contrast, each aspect references an **aspect model** that defines the structure of the aspect itself. Each aspect model represents a particular view (or aspect) of a product and is designed as a self-contained model artifact in which the nesting of further aspects is not allowed.[3] The scope of an aspect model is defined by the group of properties, events, and operations it contains, and the semantics of the properties are defined in the underlying BAMM characteristic.

Although nesting of aspect models is not supported, reuse of existing BAMM model elements is generally possible because every element in the aspect model has a unique ID within a defined namespace. It is generally allowed to reference model elements regardless of their namespace. For example, a BAMM property can refer to a BAMM characteristic defined in a different namespace. Therefore, elements from other namespaces can be mixed in one aspect model.

There are multiple approaches to structure aspect models for a specific domain. For the product modeling example from Figure 2, we have the following options:

1. Let “Product Type Data Sheet” be one aspect of the product, in which we put all properties from all the major domains, such as technical details and downloads, etc., into it. This leads to one big aspect model with many properties, characteristics, and entities.
2. Let each major domain such as technical details be one aspect of the product. This leads to several aspect models for the overall product type data sheet.
3. Let concrete sub-domains such as mechanics be one aspect of the product. This leads to many aspect models, with reusability even beyond product types.

All the different structuring approaches are valid, and it depends on the context of the aspect model's emergence which approach should be chosen. Unless the vision and scope for a new aspect model is already completely set, it is usually a good idea to start with approach 1: capture the concepts of the domain in a single model where possibilities for model element reuse quickly become apparent. Such an aspect model can be defined in a generic namespace, such as *urn:bamm:com.mycompany.mydomain:1.0.0*.

If the model grows beyond a cleanly outlined scope and describes multiple pieces of a domain that have no inherent connection (I.e., no dependency between them), a switch to approach 2 or 3 can be made by refactoring groups of elements into their own namespaces. This can lead to namespaces such as *urn:bamm:com.mycompany.mydomain.subdomain1:1.0.0*, *urn:bamm:com.mycompany.mydomain.subdomain2:1.0.0* and so on.

---

[3] This thinking is in alignment with the submodel concept of the Asset Administration Shell, the digital twin implementation of Industrie 4.0.

Finally, if reusable model elements such as properties and characteristics are organized in well-structured namespaces, a further approach can be taken: Multiple aspect models can be built that make use of the existing pool of reusable model elements, where it makes sense. Many model elements will be so specific to a certain aspect model that they cannot be reused, but whenever multiple aspect models refer to the same concept from a broader domain, reuse of an existing element should be possible.

In this whitepaper, approach 2 was chosen. This approach is also recommended by the Industrial Digital Twin Association (IDTA) for its submodel template "Generic Frame for Technical Data for Industrial Equipment in Manufacturing"[4]. It allows structuring according to the sub-domains. The next chapter shows how the aspect model “technical details” can be modeled with BAMM following approach 2.

4 Modeling product characteristics with BAMM

This chapter demonstrates how the product example illustrated in Chapter 3 can be modeled with BAMM using the Aspect Model Editor (AME)[5].

4.1 Creating the first aspect model

Figure 3 shows an aspect model skeleton for “technical details” in Figure 2. Note that AME automatically creates placeholders property1 and Characteristic1 for the aspect model. The BAMM meta-model defines metadata to the aspect model itself, such as a preferredName and a human-readable description. To exemplify the modeling capacity of BAMM, this aspect model will be extended with selected product characteristics in the following sections.

[4] https://industrialdigitaltwin.org/en/content-hub/submodels
[5] https://openmanufacturingplatform.github.io/

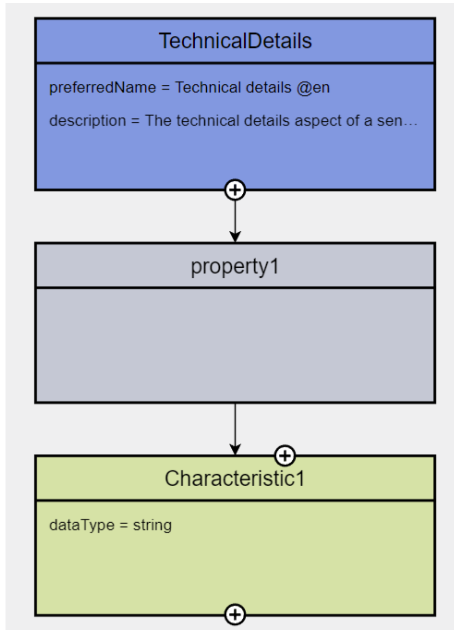


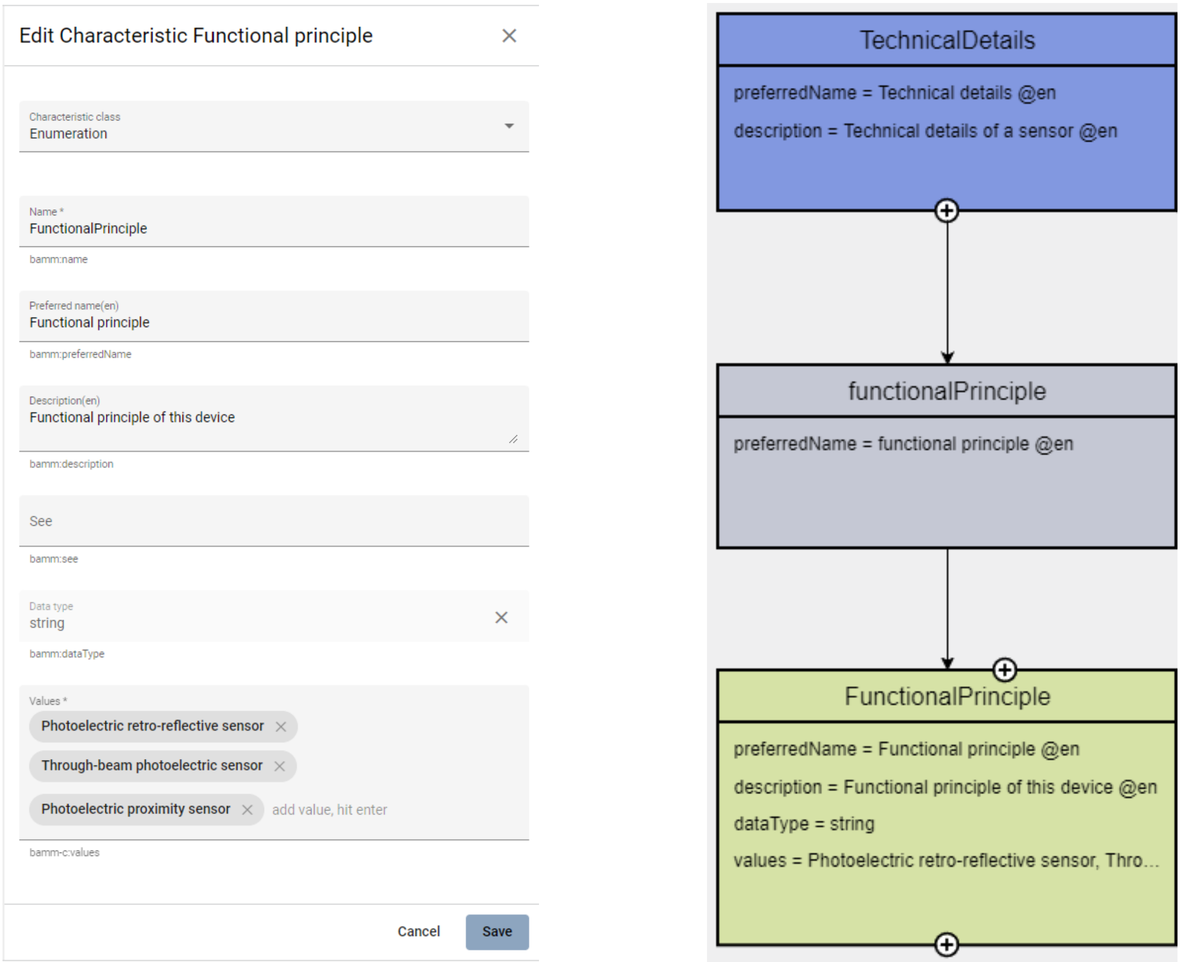
Figure 3 Basic structure of technical details from Figure 2 in AME

4.2 Characteristics with predefined value list

A common requirement for a product characteristic in the industry is a predefined value list, such as functional principle under Features in Figure 2. To model this characteristic, BAMM provides the characteristic class Enumeration[6], which takes a list of possible values. With AME, Functional principle can be modeled as an Enumeration by selecting Enumeration from the drop-down list Characteristic class, as shown in
Figure 4 (a). The individual values can be added to the Values list. Note that if the data type is defined for the enumeration, then the values need to match the definition.

The result of the model is illustrated in
Figure 4 (b). The model specifies that a TechnicalDetails aspect shall provide a value for the property functionalPrinciple (grey), which is semantically defined with the characteristic FunctionalPrinciple (green).

[6] https://openmanufacturingplatform.github.io/sds-documentation/bamm-specification/snapshot/characteristics.html#enumeration-characteristic

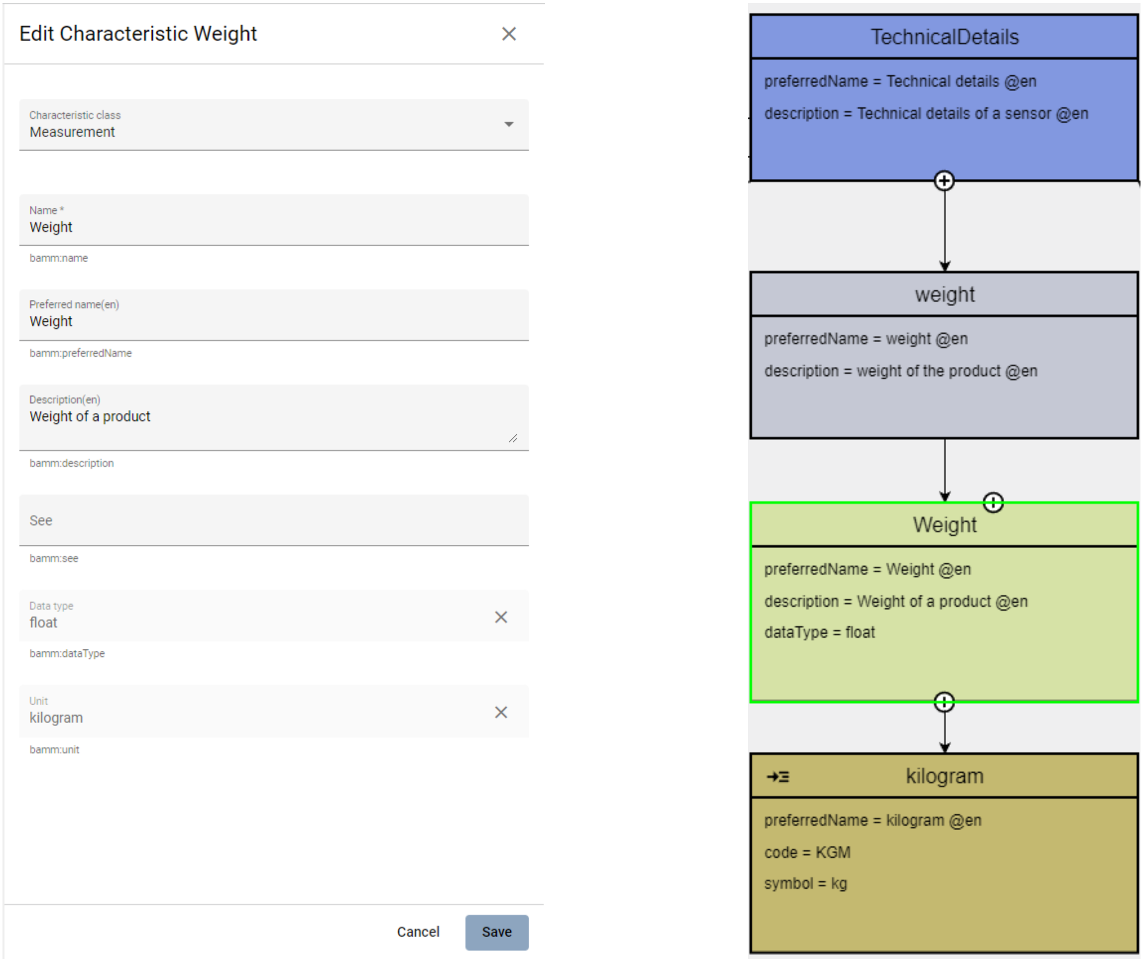


(a) (b)

Figure 4 The characteristic functional principles in AME

4.3 Characteristics of physical quantity and unit

Another common requirement in the industry sector is the modeling of physical quantities and units. Take the weight attribute from the mechanical data in Figure 2 as an example. We can create a new characteristic from the so-called Measurement class and define kilogram as the unit for it from the drop-down list. Since weight shall be numerical data, we select float as its data type, c.f. Figure 5 (a).



(a) (b)

Figure 5 The characteristic weight in AME

The result of the model is shown in Figure 5 (b). Compared with the characteristic FunctionalPrinciple in section 4.2, the model of weight has a reference to the specified unit kilogram. Information about the unit kilogram itself is covered by the BAMM meta-model and is automatically loaded into AME.

4.4 Characteristics with internal structures

In more complex situations, it is desirable to group several related information about a product into one characteristic. For example, the attribute Dimensions (W x H x D) in Figure 2 encapsulates the measurement of length in three dimensions: width, height, and depth. A client that reads this information may decide how to represent them, for example a single string from all three values in Figure 2. However, as an aspect model of the product, it is beneficial to maintain a clean structure of the three dimensions.

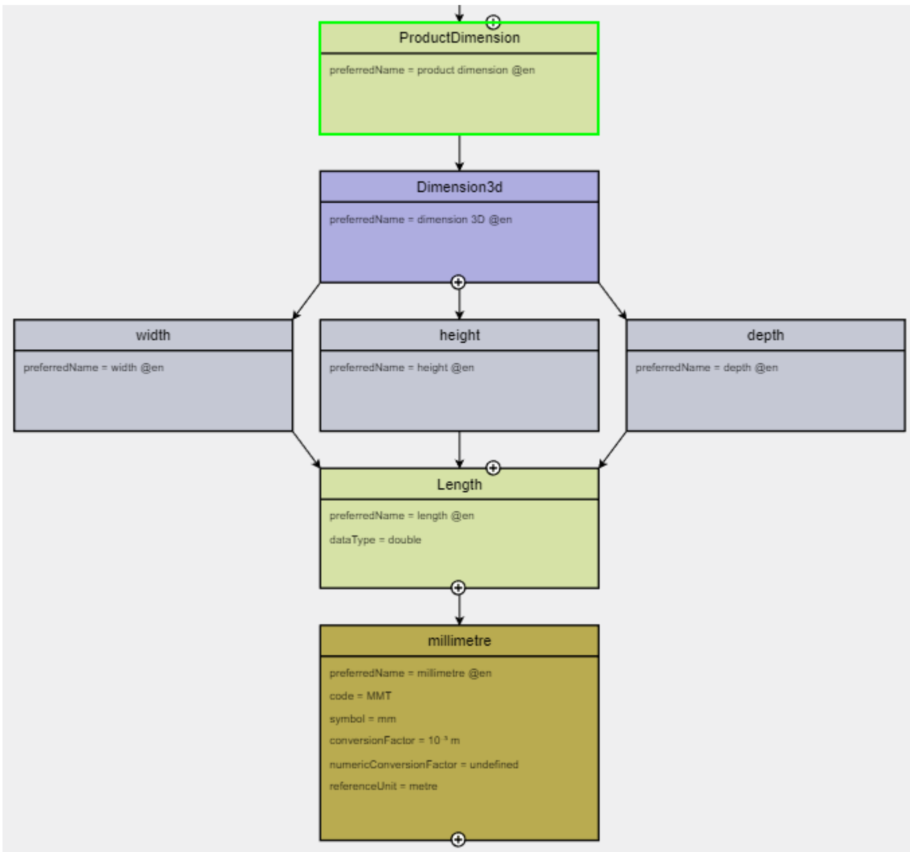
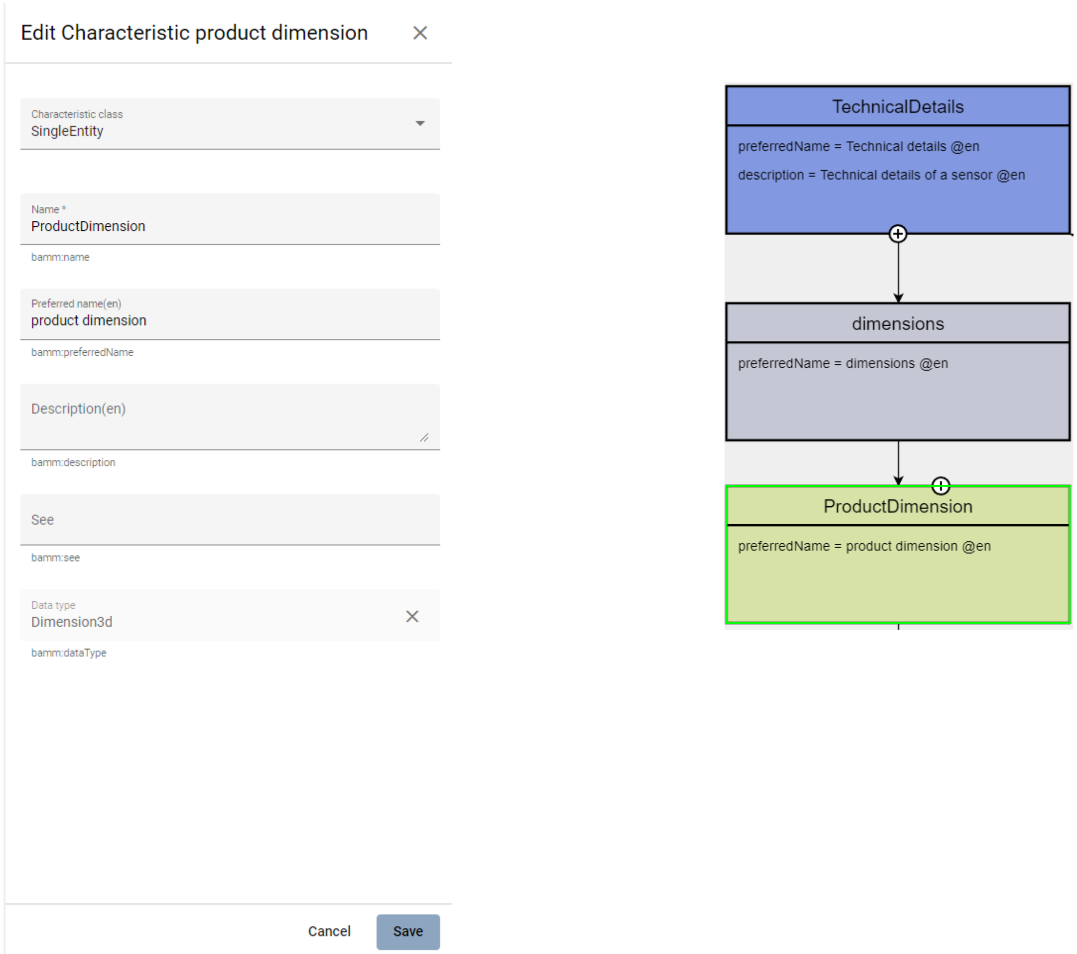


Figure 6 Structure of product dimension

The BAMM meta-model provides the concept **Entity** to model such structures. **Error! Reference source not found.** illustrates the characteristic model of ProductDimension. It is modeled as a SingleEntity characteristic whose data type is set to the entity Dimension3d. The latter one has three properties: width, height, and depth. The semantics of these properties are defined by the characteristic Length, which has been specified as a Measurement with the unit millimeter.



(a) (b)

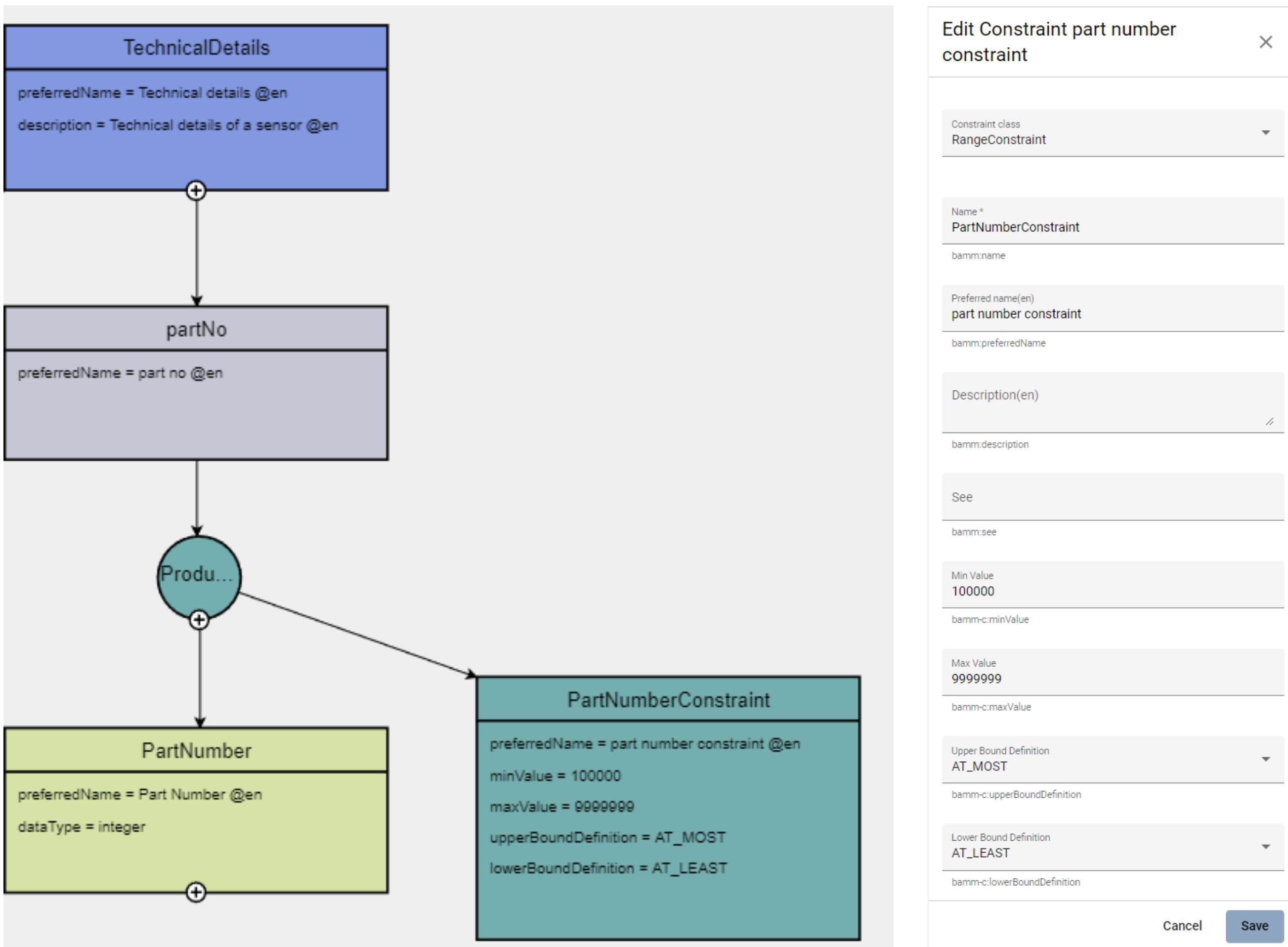
Figure **7** Product dimension in AME

After the characteristic ProductDimension is specified, as shown in Figure 7 (a), it can be attached to the aspect model via the property dimensions, as shown in Figure 7 (b).

4.5 Characteristics with value constraints

In addition to enumeration characteristics, BAMM meta-model also supports characteristics with complex value constraints, such as min and max values or regular expressions. In Figure 2, the product example has a part number (part no.) with value 1218946. Suppose that a part number

can only take a value between 1000000 and 9999999, then a corresponding characteristic can be modeled with the Range Constraint from BAMM, as shown in
Figure 8 (a).



(a) (b)

Figure 8 Part number and its constraint in AME

The concrete value range of the characteristic is defined with metadata of the constraint object (dark green), including min value, max value, upper bound definition, and lower bound definition (cf.

Figure 8 (b)). The latter two decide whether the min value and max value are inclusive or exclusive. For more details, please refer to the original BAMM specification[7]. A client that reads concrete value of a part number can check for the validity of the value using the constraint specified in the aspect model.

4.6 Collection characteristics

BAMM supports characteristics which take values in the form of collections, such as lists and sets. The product example from Figure 2 has a major domain, “downloads”, which contains various kinds of documents that can be downloaded from the web page. Figure 9 shows a snippet of the available documents.

Figure 10 shows an aspect model that describes part of the product publication documents from Figure 9. This new aspect model[8] contains three properties: product information, operating instructions, and data sheet, each of which stands for one type of document. From a data modeling perspective, all these document types shall share the same data structure, which is defined by the common characteristic ProductPublicationSet. This characteristic is modeled as a Set, whose data type is set to the entity ProductPublicationEntity. This model means that the properties product information, operating instructions, and data sheet shall take values in form of a Set, in which each element must be a ProductPublicationEntity. According to the BAMM meta-model, a set may not contain duplicates and is unordered, therefore the values of these properties must be exclusive.

---

[7] https://openmanufacturingplatform.github.io/sds-documentation/bamm-specification/snapshot/characteristics.html#range-constraint

[8] There is a submodel template "Handover Documentation" in Review at the IDTA (https://industrialdigitaltwin.org/en/content-hub/submodels). It is not the intention to propose a new submodel template for documentation in this document.

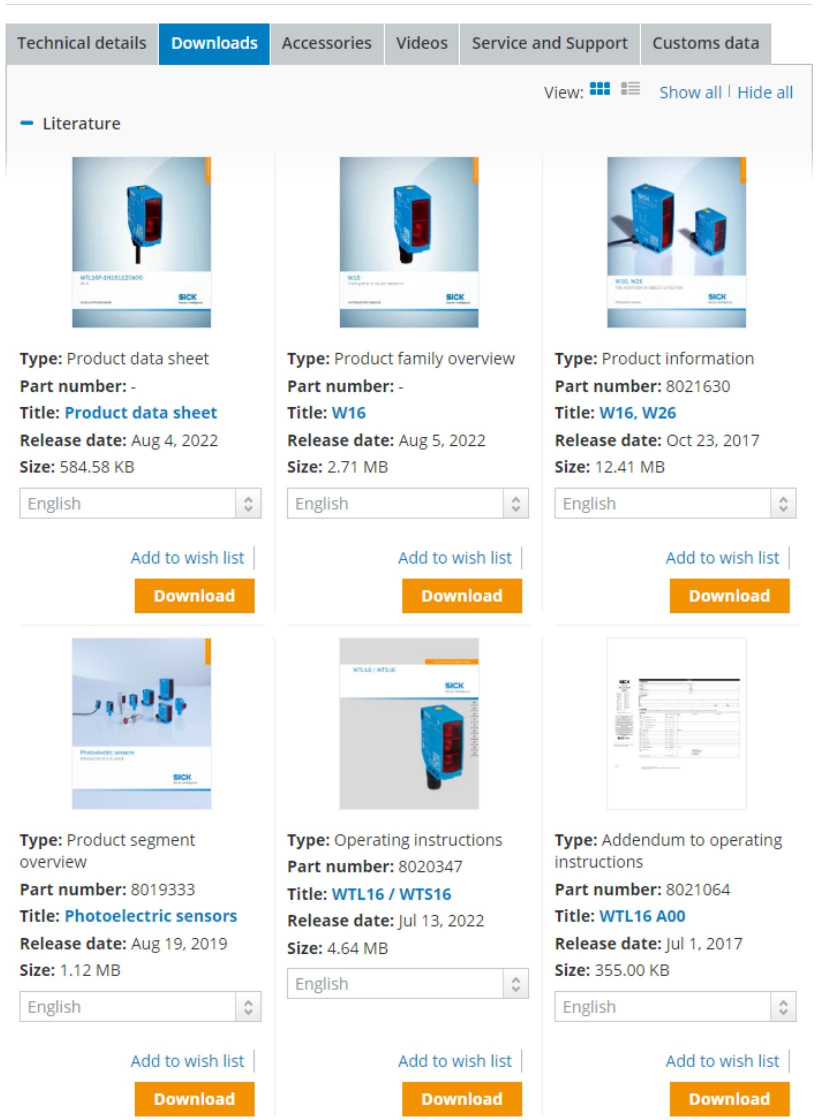


Figure 9 The major domain “downloads”

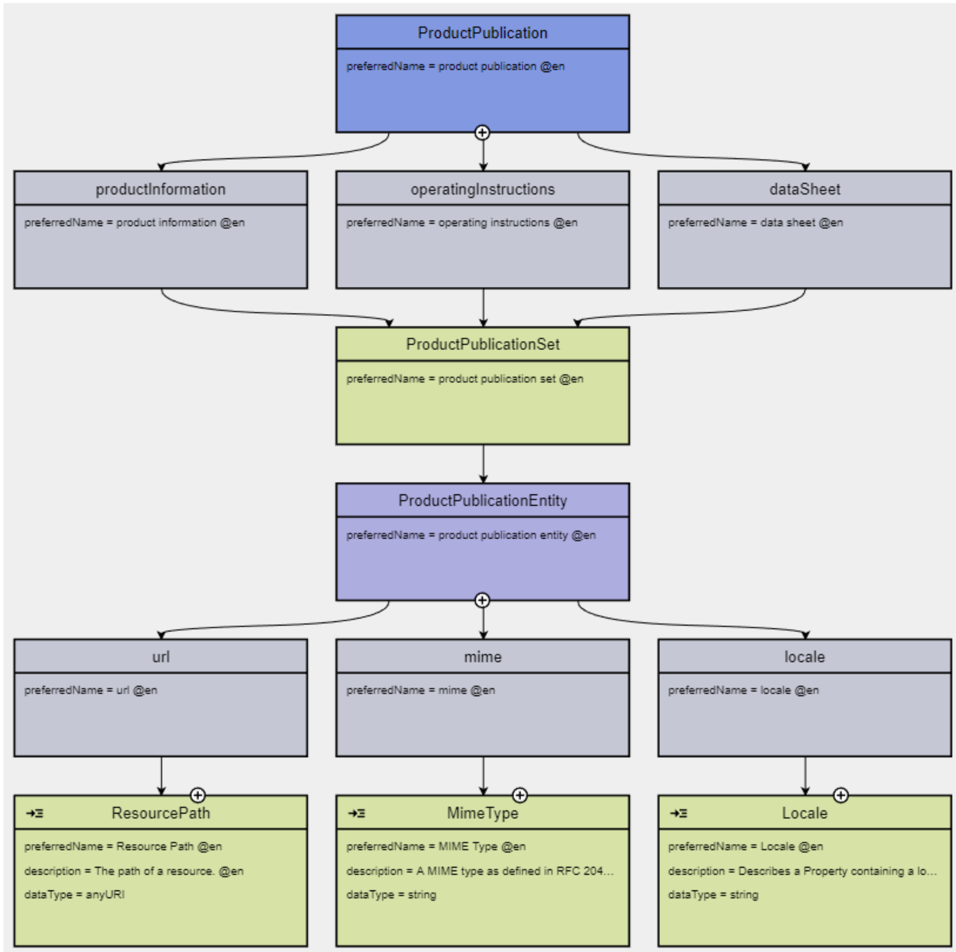


Figure 10 Aspect model product publication

The structure of a ProductPublicationEntity is defined regularly using BAMM properties and characteristics. More specifically, a ProductPublicationEntity must provide three values: a URL where the document can be found, a mime type that defines the document type, and a locale according to IETF BCP 47. Note that all these three characteristics are predefined characteristic instances by BAMM[9].

[9] https://openmanufacturingplatform.github.io/sds-documentation/bamm-specification/snapshot/characteristics.html#characteristics-instances

5 Semantics and Digital Twins

In this document, we discussed step by step how to describe the semantics of a technical description for a specific product. Figure 11 shows how the resulting semantic model can be used as a semantic description of the submodel of an Asset Administration Shell: by referencing it via its unique URN as "semanticId."

The AASX Package Explorer[10], an editor for Asset Administration Shells (AAS) as used at the bottom in Figure 11, offers a BAMM importer. This enables us to make a concrete submodel for the product represented by the digital twin. This means adding real-world values (not only example values) compliant with the semantics defined for the product. An aasx generator is also available in the command line tooling of the Java SDK[11], supporting the BAMM Aspect Meta Model.

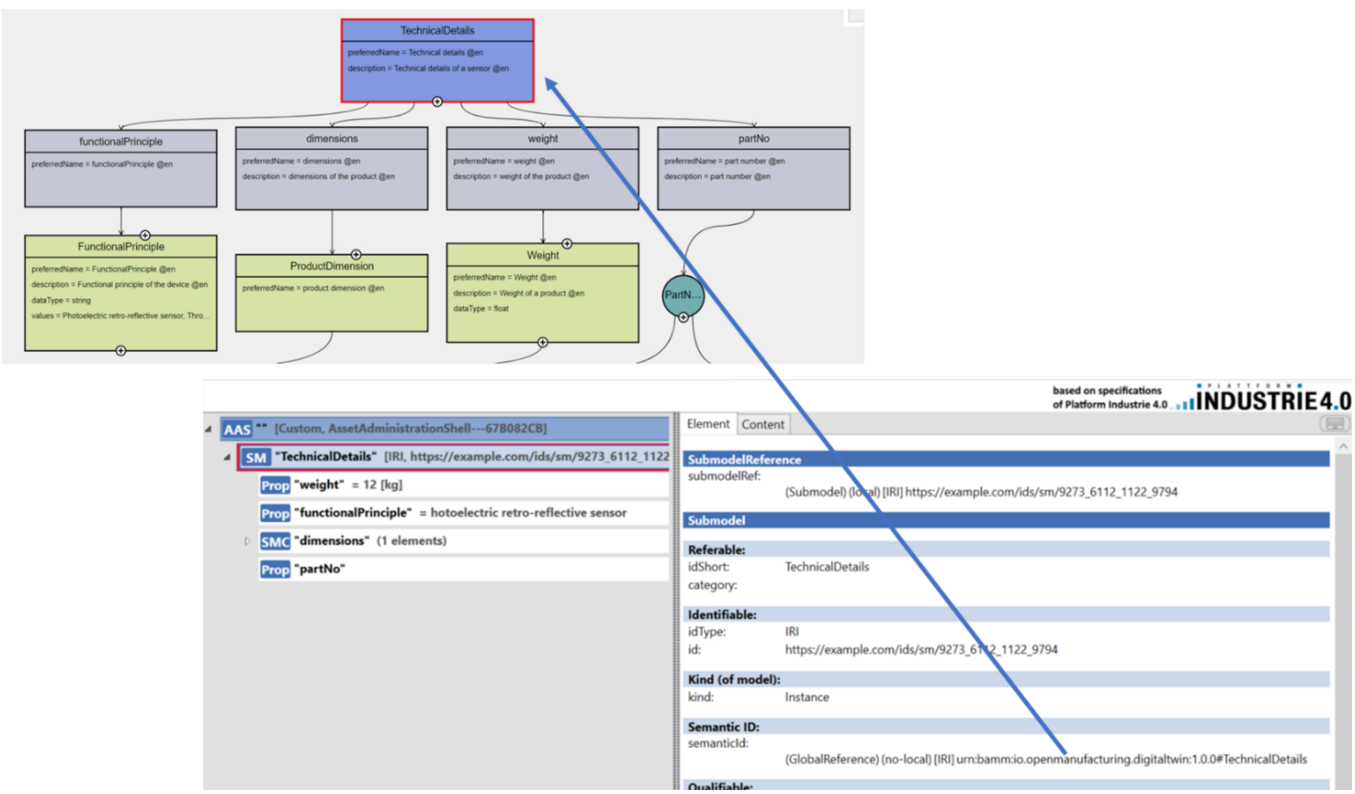


Figure 11 SemanticId of AAS mapped to aspect model

[10] https://projects.eclipse.org/projects/dt.aaspe. Note: The importer is not yet able to deal with complex semantic models.

[11] https://github.com/OpenManufacturingPlatform/sds-sdk

6 Summary and Future Work

In this whitepaper, we demonstrated the BAMM modeling capacity using real-world examples from the industry sector. We discussed how an aspect model can be created using the Aspect Model Editor and how concrete product characteristics can be modeled.

The BAMM meta-model has rich semantics, enabling it to cover the typical modeling requirements such as a predefined value list (section 4.2), physical quantities and units (section 4.3), characteristics with internal structures (section 4.4), characteristics with value constraints (section 4.5), and collection characteristics (section 4.6). We also discussed building a holistic aspect model by combining several reusable model elements (section 4.7). Aspect Models can be used to describe the semantics of submodels of the Asset Administration Shell, the implementation of digital twins of Industry 4.0 (section 5).

It is worth mentioning that the Aspect Model Editor (AME) plays a vital role while working with BAMM. It guides the user through the modeling process and provides model examples and built-in features that intuitively allow the creation of aspect models. It also hides the abstract concepts of RDF, making BAMM accessible to domain experts.

We plan to use BAMM for more complex modeling tasks in future work. For example, user-defined characteristics and constraint classes may be necessary for specific use cases that require highly customized data models. The Aspect Model Editor can also be extended with more convenient features that further improve the modeling experience.