

IDTA 02005-1-0

Provision of Simulation Models

December 2022

SPECIFICATION

Submodel Template of the
Asset Administration Shell



Submodel Template

IDTA **approved**

- 100% AAS compliant
- Consistent & interoperable
- Released by the AAS experts

Imprint

Publisher

Industrial Digital Twin Association
Lyoner Strasse 18
60528 Frankfurt am Main
Germany
<https://www.industrialdigitaltwin.org/>

Version history

Date	Version	Comment
15.12.2022	1.0	Release of the official Submodel template published by IDTA.

Contents

1	General	7
1.1	About this document	7
1.2	Scope of the Submodel	7
1.3	Relevant standards for the Submodel	7
1.3.1	AAS Submodel “Contact Information”	7
1.3.2	MIC (Model identity Card)	8
1.3.3	FMI (Functional Mockup Interface)	8
1.3.4	ECLASS	8
1.3.5	AutomationML	8
1.4	Future relevant standards	9
1.4.1	SSP (System Structure and Parameterization)	9
1.4.2	ProStep iViP Project Smart Systems Engineering	9
1.4.3	AAS Submodel candidate “Software Nameplate”	9
1.4.4	AAS Submodel candidate “Hierarchical Structures enabling Bills Of Material”	10
1.4.5	Spatially discretizing methods	10
1.4.6	Quality of Simulation Model	10
1.5	Use cases, requirements and design decisions	10
1.5.1	Use case 1: Provision of simulation models to asset types	12
1.5.2	Use case 2: Provision of simulation models to asset instances	12
1.5.3	Use case 3: Create and maintain a simulation of Systems	12
1.5.4	Requirements	12
1.5.5	Design decisions	13
2	Submodel “Provision of Simulation Models”	15
2.1	Approach	15
2.2	Properties of the Submodel “Provision of Simulation Models”	15
2.3	Attributes of Submodel instance	16
2.4	SubmodelElements of SimulationModel	17
2.5	SubmodelElements of SimPurpose	19
2.6	SubmodelElements of Environment	20
2.7	SubmodelElements of SimulationTool	21
2.8	SubmodelElements of SolverAndTolerances	22
2.9	SubmodelElements of TestedToolSolverAlgorithm	23
2.10	SubmodelElements of ModelFile	24
2.11	SubmodelElements of ModelFileVersion	25
2.12	SubmodelElements of SimModManufacturerInformation	26

2.13	SubmodelElements of Email.....	27
2.14	SubmodelElements of Phone	28
2.15	SubmodelElements of Ports	29
2.16	SubmodelElements of PortsConnector	30
2.17	SubmodelElements of Variable	31
2.18	SubmodelElements of BinaryConnector.....	32
2.19	Predefined values for properties.....	32
2.19.1	Value list for PosSimPurpose and NegSimPurpose.....	33
2.19.2	Value list for TypeOfModel	33
2.19.3	Value list for ScopeOfModel	33
2.19.4	Value list for LicenseModel.....	34
2.19.5	Value list for EngineeringDomain	34
2.19.6	Value list for VisualizationInformation.....	35
2.19.7	Value list for ParamMethod	35
2.19.8	Value list for InitStateMethod.....	35
2.19.9	Value list for VariableType.....	36
2.19.10	Value list for VariableCausality	36
2.19.1	Value list for VariablePrefix.....	36
3	AAS SM Modeling Scenarios.....	37
3.1	Managing multiple models.....	37
3.1.1	Explanatory example:	37
3.1.2	Recommended modeling:.....	37
3.1.3	Simulation Model Bugfix	38
3.1.4	New Simulation Model	39
3.2	Model requests to component supplier.....	39
3.2.1	Offering explicit models	40
3.2.2	Query of individual model	40
3.3	Modeling Supported Environments	41
Annex A.	Explanations on used table formats	44
1.	General	44
2.	Tables on Submodels and SubmodelElements.....	44
	Bibliography	45

Figures

Figure 1: Value network adaptable factory [1]	11
Figure 2: Usecases for AAS Submodel “Provision of Simulation Models”	11
Figure 3: Information model architecture with the first available Submodels	13
Figure 4: Mapping of simulation models in the machine hierarchy	14
Figure 5: Using Submodel “Hierachical Structures enabling Bills of Material” for assembly of simulation model	14
Figure 6: UML Diagram of the Submodel “Provision of Simulation Models”	16
Figure 7: Structure and choronological development of the simulation models.	37
Figure 8: Modeling approaches for the Submodel “Provision of Simulation Models”	38
Figure 9: Various environments, represented by operating systems and simulation tools	41
Figure 10: Simulation tools with compiler and solvers	42

Tables

Table 1: Requirements to the Submodel "Provision of Simulation Models"	12
Table 2: Attributes of Submodel instance	16
Table 3: Submodel elements of SimulationModel	17
Table 4: Submodel elements of SimPurpose	19
Table 5: Submodel elements of Environment.....	20
Table 6: Submodel elements of SimulationTool.....	21
Table 7: Submodel elements of SolverAndTolerances	22
Table 8: Submodel elements of TestedToolSolverAlgorithm	23
Table 9: Submodel elements of ModelFile	24
Table 10: Submodel elements of ModelFileVersion	25
Table 11: Submodel elements of SimModManufacturerInformation	26
Table 12: Submodel elements of Email.....	27
Table 13: Submodel elements of Phone	28
Table 14: Submodel elements of Ports	29
Table 15: Submodel elements of PortsConnector.....	30
Table 16: Submodel elements of Variable.....	31
Table 17: Submodel elements of BinaryConnector	32
Table 18: Value list for PosSimPurpose and NegSimPurpose.....	33
Table 19: Value list for TypeOfModel	33
Table 20: Value list for ScopeOfModel	34
Table 21: Value list for LicenseModel.....	34
Table 22: Value list for EngineeringDomain	34
Table 23: Value list for VisualizationInformation	35
Table 24: Value list for ParamMethod	35
Table 25: Value list for InitStateMethod.....	35
Table 26: Value list for VariableType.....	36
Table 27: Value list for VariableCausality.....	36
Table 28: Value list for VariablePrefix	36

1 General

1.1 About this document

This document is a part of a specification series. Each part specifies the contents of a Submodel for the Asset Administration Shell (AAS). The AAS is described in [1], [2], [3] and [6]. First exemplary Submodel contents were described in [4], while the actual format of this document was derived by the "Administration Shell in Practice" [5]. The format aims to be very concise, giving only minimal necessary information for applying a Submodel template, while leaving deeper descriptions and specification of concepts, structures and mapping to the respective documents [1] to [6].

The target group of the specification are developers and editors of technical documentation and manufacturer information, which are describing assets in smart manufacturing by means of the Asset Administration Shell (AAS) and therefore need to create a Submodel instance out of a Submodel template with a hierarchy of SubmodelElements. This document especially details on the question, which SubmodelElements with which semantic identification shall be used in the Submodel template.

1.2 Scope of the Submodel

The presented Submodel allows the interoperable provision of simulation model files for an asset via the asset administration shell. The Submodel enables this for a wide range of simulation model types and simulation purposes. It contains information about the type of model, how to use the model, and the areas of application.

The main underlying class of models for this Submodel are DAEs, models described by differential algebraic equations. However, models of other types, such as CAD, FMEA, etc., can also be described. FEM type models are not considered explicitly, they will be covered by another Submodel by interest. The models described by this Submodel may be provided in ASCII or binary form to be used with a specific simulation software (e.g. Matlab/Simulink, Virtuos, etc.), as source code (e.g. C, Modelica, etc.), or in a model exchange format such as FMI.

It describes rudimentarily the content of the model.

Assets, where you can provide simulation models via AAS, can be passive parts, active devices, subsystems, machines or even production plants. When using this Submodel template, it is required that the asset for which a simulation model should be provided has an AAS. That is, if there are no AAS yet, first an asset-ID needs to be defined and an AAS created. A Submodel can then be added to this AAS.

In the first step, the Submodel supports searching and finding, as well as requesting simulation model files from manufacturers or distributors.

In further steps, the Submodel will also support the automatic integration of a model into a specific simulation environment, up to an automatic generation of an overall simulation based on structural descriptions of a system containing different components. As it is possible with e.g., the Submodel candidate for hierarchical structures enabling "Bills of Material" (BoM).

1.3 Relevant standards for the Submodel

Here standards are listed which have an influence on this version of the Submodel or which are referenced.

In addition, some important candidates for further development are listed.

1.3.1 AAS Submodel "Contact Information"

In order to provide contact options to a simulation-specific support or for the request of a simulation model, structures and features were used from the Submodel "Contact Information" (IDTA 02002-1-0).

The name of the provider of the simulation model, the available communication languages and optionally an e-mail address or a telephone number can be specified.

1.3.2 MIC (Model identity Card)

The Model Identity Card (MIC) contains information to characterize simulation models based on a set of core information [10]. It consists of definitions for general information, integration, content and computation, ports, internal variables and parameters, verification and validation. Information about the port definition were used as inspiration for the Submodel template.

1.3.3 FMI (Functional Mockup Interface)

The Functional Mock-up Interface (FMI) is a free standard for exchange and co-simulation of dynamic models. It defines a model container and a model interface using a combination of XML files, binaries and / or C code zipped into a single file. An integration algorithm may be included for co-simulation of models. It is supported by [170+](#) tools and maintained as a [Modelica Association Project \[8\]](#) on [GitHub \[9\]](#). Since a large number of simulation models are already provided as FMUs and will continue to be so in the future, the Submodel “Provision of Simulation Models” attempts to represent this type of model as accurately as possible.

1.3.4 ECLASS

ECLASS defined tens of thousands product classes and unique properties to standardize procurement, storage, production, and distribution within and between companies across different business areas, countries, and languages. ECLASS is the only ISO/IEC conform industry standard for classification and unique description of products and services. These unique identifiers called IRDIs (International Registration Data Identifiers) are based on ISO/IEC 11179-6, ISO 29002 and ISO 6532 and are used by AAS to provide data specifications for its properties, values, and units. ECLASS was selected by the IDTA as one of the preference standards to provide data specifications. These unique identifiers called IRDIs (International Registration Data Identifiers) are based on ISO/IEC 11179-6, ISO 29002 and ISO 6532 and are used by AAS to provide data specifications for its properties, values, and units. ECLASS was selected by the IDTA as one of the preferred standards to provide data specifications.

1.3.5 AutomationML

AutomationML [\[11\]](#) is a comprehensive XML based object-oriented data modeling language. It allows the modeling, storage and exchange of engineering models covering a multitude of relevant aspects of engineering. AML provides a comprehensive set of flexible mechanisms and innovations to model today's engineering aspects as well as future engineering aspects to come. Its language characteristics allow to model existing or new domain models.

AutomationML has a lean and distributed file architecture. It does not define any new file format but combines existing established XML data formats which have been proven in use for their specific domain. This is why the normative part of the IEC62714-1:2018 document consists of 32 pages only. The data formats for the following modeling domains are:

- object topologies including hierarchies, properties and relations of objects: CAEX according to IEC 62424
- geometries and kinematics of objects: COLLADA 1.4.1 and 1.5.0 (ISO/PAS 17506:2012)
- discrete behavior of objects: PLCopen XML 2.0 and 2.0.1; in addition, IEC62714-4 will allow the usage of IEC61131-10

AutomationML also provides an [application recommendation \[12\]](#) which describes the serialization of the AAS Metamodel using AutomationML by defining appropriate serialization rules and role, interface and system unit classes related to them.

1.4 Future relevant standards

The initial version will contain all relevant properties relevant for the use cases explained in this document. For future releases we are also evaluating other standardization activities. Examples include current releases of the VDI GMA FA 6.XX about quality metrics but also other IDTA working group draft generic forms like the Submodel “Hierarchical Structures enabling Bills of Material”.

1.4.1 SSP (System Structure and Parameterization)

SSP is a tool-independent format developed by the [Modelica Association Project \[8\]](#) to exchange simulation-capable system structures consisting of individual simulation components, preferably FMUs (Functional Mockup Units). With SSP such a system of interlinked components can be described by a well-structured bundle of subformats, packed into a zip-container. The package comprises the specification of the hierarchical and functional structure of the component network (including all signal flows), the parameters, parameter sets and parameter mappings as well as dictionaries of signals. The first version of SSP is already supported by some system simulation tools (<https://ssp-standard.org/tools/>). The format can be of interest to the Submodel as it can serve as specification and exchange format for the Use Case 3 “Generate and Simulate a system simulation out of existing simulation components”.

1.4.2 ProStep iViP Project Smart Systems Engineering

The [Smart Systems Engineering \(SmartSE\)](#) project group works together with a group of participants from almost 30 companies and research institutions to develop application-oriented concepts for overcoming common systems engineering challenges. Since 2012 SmartSE develops recommendations as well as promotes and industrializes technical standards with regard to collaborative systems engineering tasks within and especially across companies. The latest recommendations apply to [Simulation-based decision making](#) and the format [System Structure and Parameterization](#). A general SmartSE recommendation V3.0 is in work ([SmartSE V2.0](#)). Despite the slightly different focus of system simulation models (SmartSE more focusing on individual models, the Submodel focusing on catalogue models) an exchange between the Submodel and the SmartSE group could yield in a review and industrialization of the Submodel specification and an extension of its current use cases.

1.4.3 AAS Submodel candidate “Software Nameplate”

The Submodel candidate “Software Nameplate” defines properties relevant for identification of software products (type) and their installed instances. This information shall be provided in a consistent manner in form of a “nameplate for software”, derived and specialized from the Submodel “Digital Nameplate for Industrial Equipment”. The nameplate for software applies to stand-alone software assets, as well as to software as integral part of a physical asset, e.g. firmware.

It describes information on

- manufacturer (name, product description, product family)
- version (version number, version name, version information)
- release date and release notes
- build date
- serial number and instance name
- installation details (installed version, installation date, path, and source)
- environment (architecture, operating system, host).

Further Submodels describing license information, software requirements, and dependencies are currently in initial discussions.

This, at the time of this specification, future Versions of the Submodel “Software Nameplate” will identify and describe a simulation model, in principle. It can be integrated in the management shell of the simulation model itself. The Submodel “Provision of Simulation Models” focuses on the description of simulation-specific properties and is integrated in the asset administration shell of the asset itself, analogous to a documentation or a CAD model for a component.

1.4.4 AAS Submodel candidate “Hierarchical Structures enabling Bills Of Material”

A Submodel that can describe the structure of an asset or an assembly. At the time of publication of the first version of the Submodel “Provision of Simulation Models”, the Submodel Hierarchical Structures enabling Bills Of Material is still work in progress.

The description primarily includes a structured "bill of material" and may also include the connections between components.

Since this Submodel is a very useful addition in perspective for simulation usecases, such as generating or assisted integration of simulation models, assumptions have been made for the coexistence of the models in the Design Decisions and Usecases chapter.

1.4.5 Spatially discretizing methods

In the current version of the Submodel, due to a lack of expertise, the needs of simulation models from the areas of finite element method (FEM) simulations were not specifically addressed. Nevertheless, simulation models from this area should be provided with the Submodel to at least represent the existence of a corresponding model within the AAS.

The potential deviations of the provided simulation model to the customer specific requirements and associated changes in the model can be overcome by providing multiple parameter files and/or providing multiple complete simulation models. Furthermore, the Submodel with the simulation documentation offers the possibility to formulate a detailed description about the application area of a simulation model.

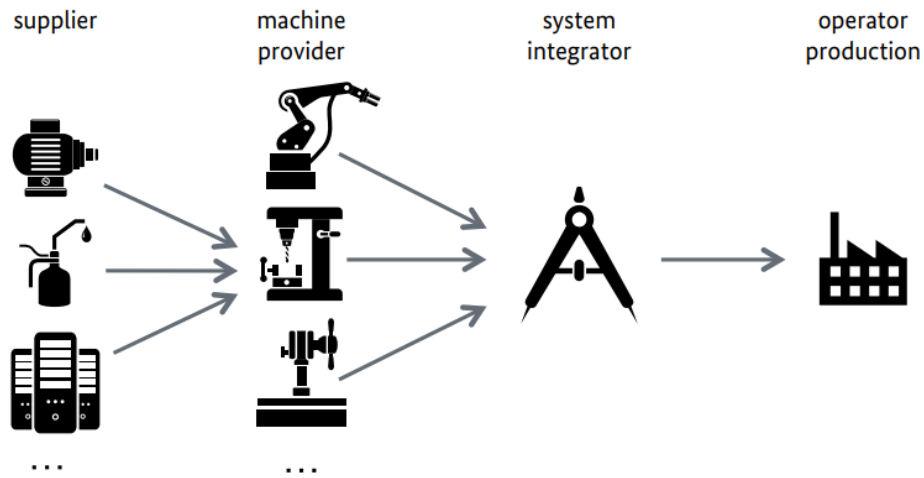
In future versions of the Submodel, the structure is to be adapted by the inclusion of creators and users of FEM models in such a way that FEM models can also be adequately represented. Nevertheless, it should be noted that such simulations are mainly used for dimensioning components during their development and are less likely to find application in system integration. For the system integrator, a digital data sheet will be of more use. Therefore, cross-company use of such simulation models is considered less likely.

1.4.6 Quality of Simulation Model

In the working group 6.11 Virtual Commissioning of the GMA, a methodology is currently being developed which is intended to enable the quality of a simulation model to be determined in a standardized manner [(Automation) 7]. This involves evaluating a simulation model on the basis of 25 attributes, among others, and determining an overall quality for the simulation model. In the version of the Submodel described here, this quality metric is not yet applied, but this is explicitly aimed at for future versions. Possibilities for integration and collaboration are currently being discussed in talks between the working groups.

1.5 Use cases, requirements and design decisions

The Submodel “Provision of Simulation Models” can be used to implement various use cases in which simulation models are to be integrated. Three basic ones are shown in the following subchapters. All of them have in common that one is in a value network with many partners, as it was shown in Figure 1.

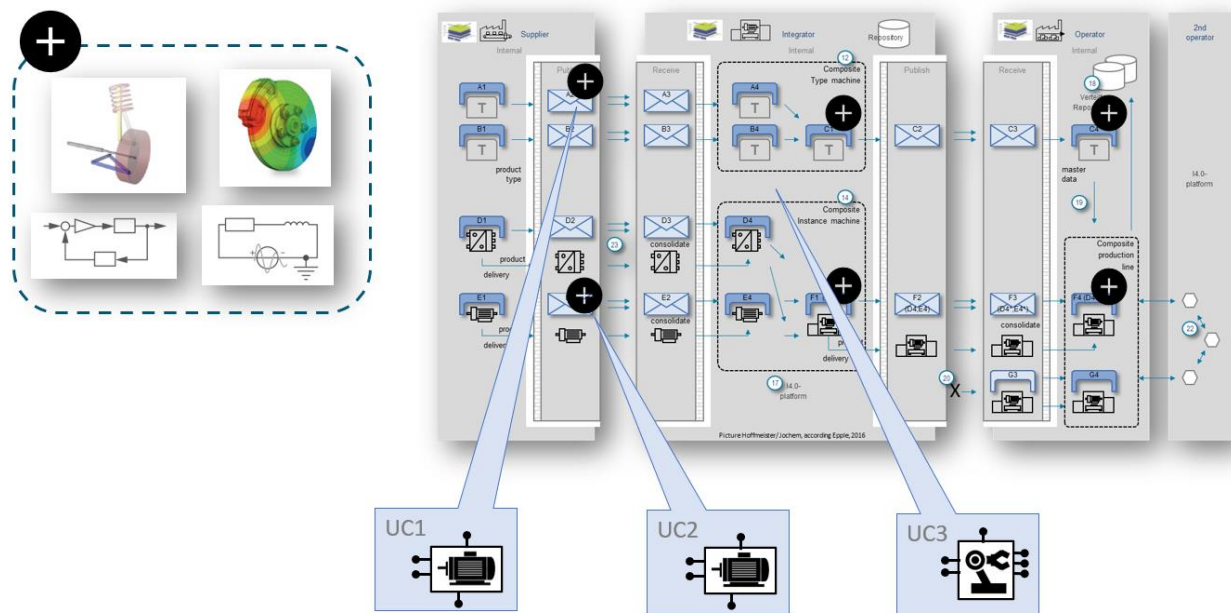
Figure 5: The “Adaptable Factory” value network

Source: Plattform Industrie 4.0

Figure 1: Value network adaptable factory [1]

To provide an overview of the scenarios were targeted with the first version of the Submodel, the use cases are located in Figure 2 of the generic value network published via Plattform Industrie 4.0.

These use cases are described in more detail in the following chapters.

**Figure 2: Usecases for AAS Submodel “Provision of Simulation Models”**

Zero to many simulation models can be provided to an asset type or instance. More than one model is needed for different simulation methods (see picture left), simulation tools and simulation purposes.

Before describing the different use cases, the differences between a product type and a product instance should be briefly discussed here.

An asset type is a representation that defines all the usual properties that are also contained in asset instances at the beginning of their life cycle [Details of the AAS Part 1]. These include, for example, the structure and components used. But also, properties resulting from the engineering are included in the type.

For example, the maximum load capacity, allowed environmental conditions. In contrast to this an instance represents a concrete and clearly identifiable entity of a certain type [DotAAS 1]. This instance can contain additional properties like color, serial number, runtime information, etc. in addition to the properties contained in the type. Furthermore, properties in the instance can change over the life cycle and thus deviate from the type.

1.5.1 Use case 1: Provision of simulation models to asset types

A user is interested in a product (type) and is offered various models via the Submodel with which the user can test the model in simulation environments.

The simulation model is typically provided by the component manufacturer.

1.5.2 Use case 2: Provision of simulation models to asset instances

A user has ordered an asset and is offered simulation models via the Submodel which he can use to simulate and test the specific behaviour of the component after integration in his own solution.

The instance simulation model differs in detail from those of a type simulation model. It can be adapted e.g., due to measured properties in production, aging phenomena in operation or replacement of subordinate components compared to an original machine. This model is therefore not necessarily provided by the manufacturer of the asset.

1.5.3 Use case 3: Create and maintain a simulation of Systems

A user is designing a solution using various assets, from different manufacturers or internal suppliers. Via the Submodel the user gets an overview which simulation models are available to realize a complete simulation of the system. If necessary, the user can send specific requests to the manufacturers/supplier of the components on the basis of the Submodel in order to obtain the corresponding models.

It also supports automated updating of models. Notifications can be generated for new simulation models for the component or new versions of a used model.

More use cases are under discussion, but current work focus on the above three use cases [13].

1.5.4 Requirements

The table describes the main requirements to the Submodel that were considered during the elaboration.

Table 1: Requirements to the Submodel "Provision of Simulation Models"

No.	Title	Description
Req 000	Providing simulation models to a component	The Submodel provides simulation models and their descriptions of an Industrie 4.0 component.
Req 010	All types of simulation models	The Submodel can represent all types of simulation models.
Req 020	Integration of standards	The Submodel integrates existing standards, such as FMI.
Req 021	Adopting artifacts	The Submodel adopts artifacts from existing standards that support the intended use cases of the Submodel and whose use also appears to make sense beyond the standard.
Req 030	Version management	The Submodel should support common version management.
Req 040	Providing simulation model files	The Submodel can make simulation model files available, such as download.
Req 041	User specific request	The Submodel is intended to enable a request for a user specific simulation model to be submitted to the component suppliers.
Req 042	Models available on request	The Submodel shall enable the supplier of a component to describe a model which can be realized and delivered.
Req 050	simulation specific properties	The Submodel should show simulation-specific properties, which support the search and comparison of suitable models.

1.5.5 Design decisions

When designing the Submodel "Provision of Simulation Models", the following specifications were made, which are shown in Figure 3.

- The Submodel "Provision of Simulation Models" complements the description of an asset
- A simulation model will have its own asset administration shell in the future, the following general Submodels can be used then
- Considering Submodels like "Digital Nameplate for Industrial Equipment", "Generic Frame for Technical Data for Industrial Equipment in Manufacturing" and "Handover Documentation"
- The Submodel "Provision of Simulation Models" can be assigned to an asset type or an asset instance
- The Submodel "Hierarchical Structures enabling Bills of Material" describes composite solutions including their connections

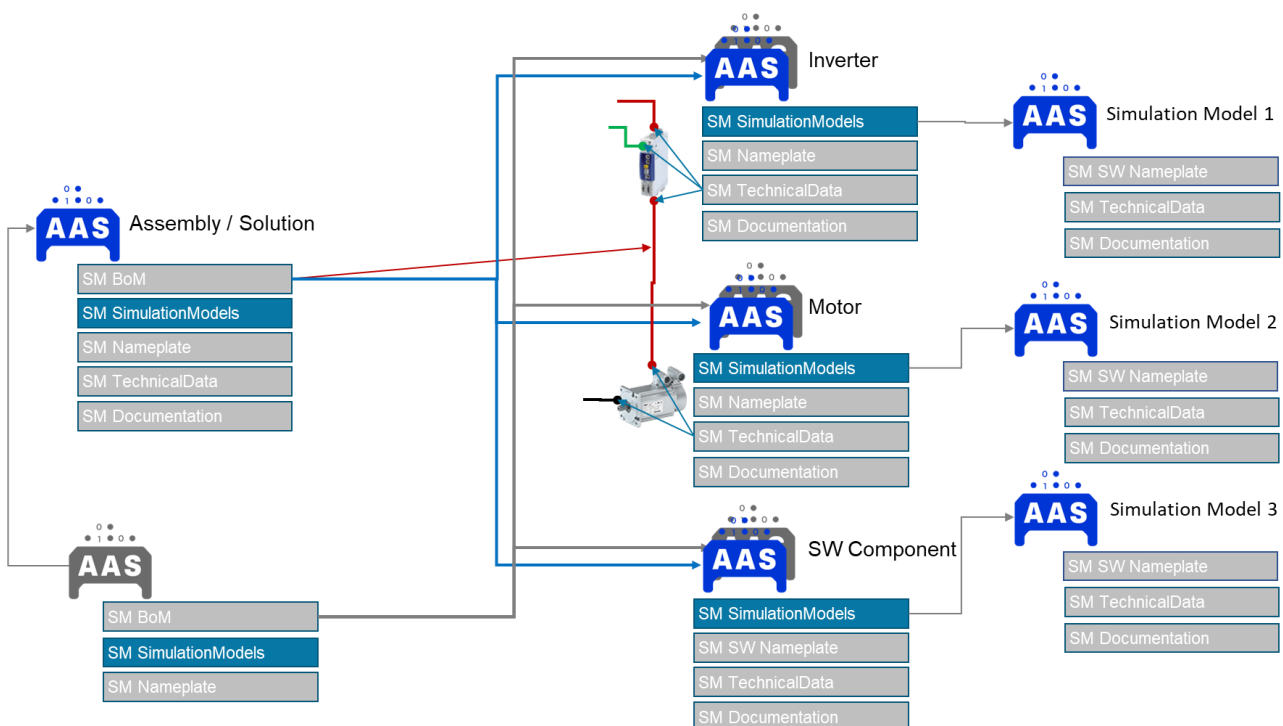


Figure 3: Information model architecture with the first available Submodels

Figure 4 shows an example of how the asset administration shell with its Submodels and simulation models can describe an asset by using the Submodel "Hierarchical Structures enabling Bills of Material" Submodel. The template specification of the Submodel "Hierarchical Structures enabling Bills of Material" is currently under construction.

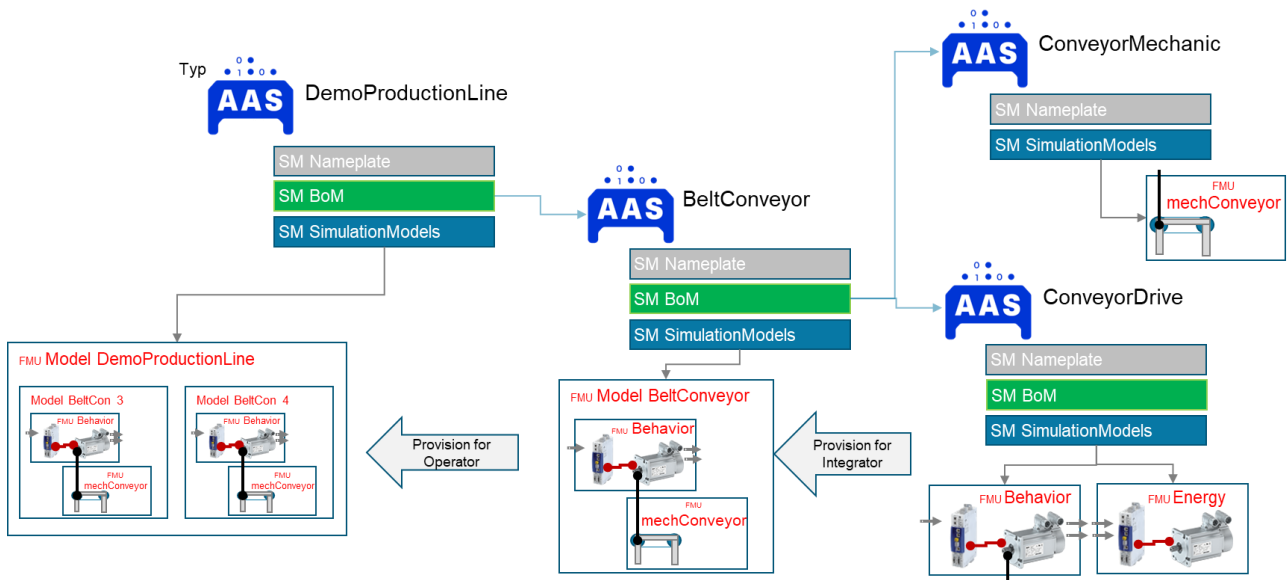


Figure 4: Mapping of simulation models in the machine hierarchy

Figure 5 shows the extended use case in which the Submodel “Hierarchical Structures enabling Bills of Material” describes the structure of a simulation model consisting of multiple sub-simulation models. The main simulation model is referred to as a gray box model, since the model consists of black box models and its superordinate structure is described via the management shell.

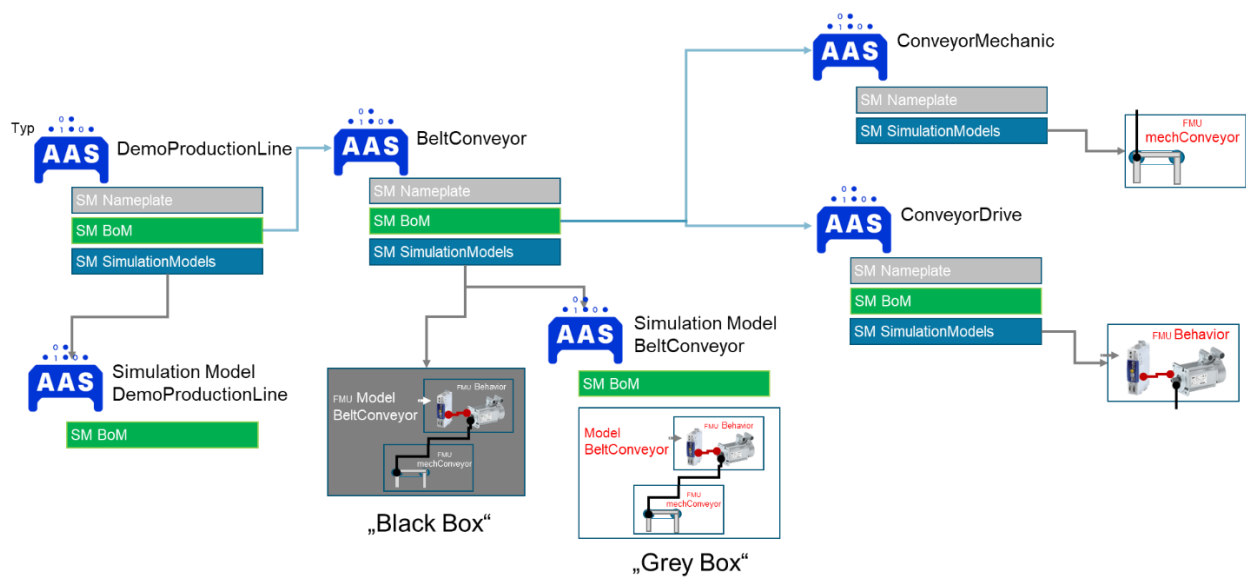


Figure 5: Using Submodel “Hierarchical Structures enabling Bills of Material” for assembly of simulation model

2 Submodel “Provision of Simulation Models”

2.1 Approach

In this document one Submodel for provision of multiple simulation model files to a component is defined. Simulation models can be added to an asset type and an asset instance.

The goal is to provide all types of simulation models for diverse simulations. With the Submodel “Provision of Simulation Models” information can be provided simplifying searching and finding of suitable models and their integration into a simulation environment. Further a standardized mechanism to find contact information to the simulation model provider is given.

In addition to the use case of providing models, the Submodel can also be used to send a specific request for a simulation model to the manufacturer/supplier of a component. For this purpose, parts of the Submodel “Contact Information” are transferred to this Submodel.

2.2 Properties of the Submodel “Provision of Simulation Models”

As an overview, the features and collections are shown here as in a UML diagram.

The following main aspects can be described with the model.

- Model file deployment with different versions
- Manufacturer's simulation support contact information
- Simulation purposes, positive and negative
- Documentation of example simulations
- Environment, and tested constellations simulation tool and solver types
- Setting options for model parameters and initial values
- License types
- Model scope, type and integrated engineering domain
- Model ports and interfaces, e.g., for visualization

Practical guidance for Submodel modelers on various scenarios of mapping is presented in Chapter 3.

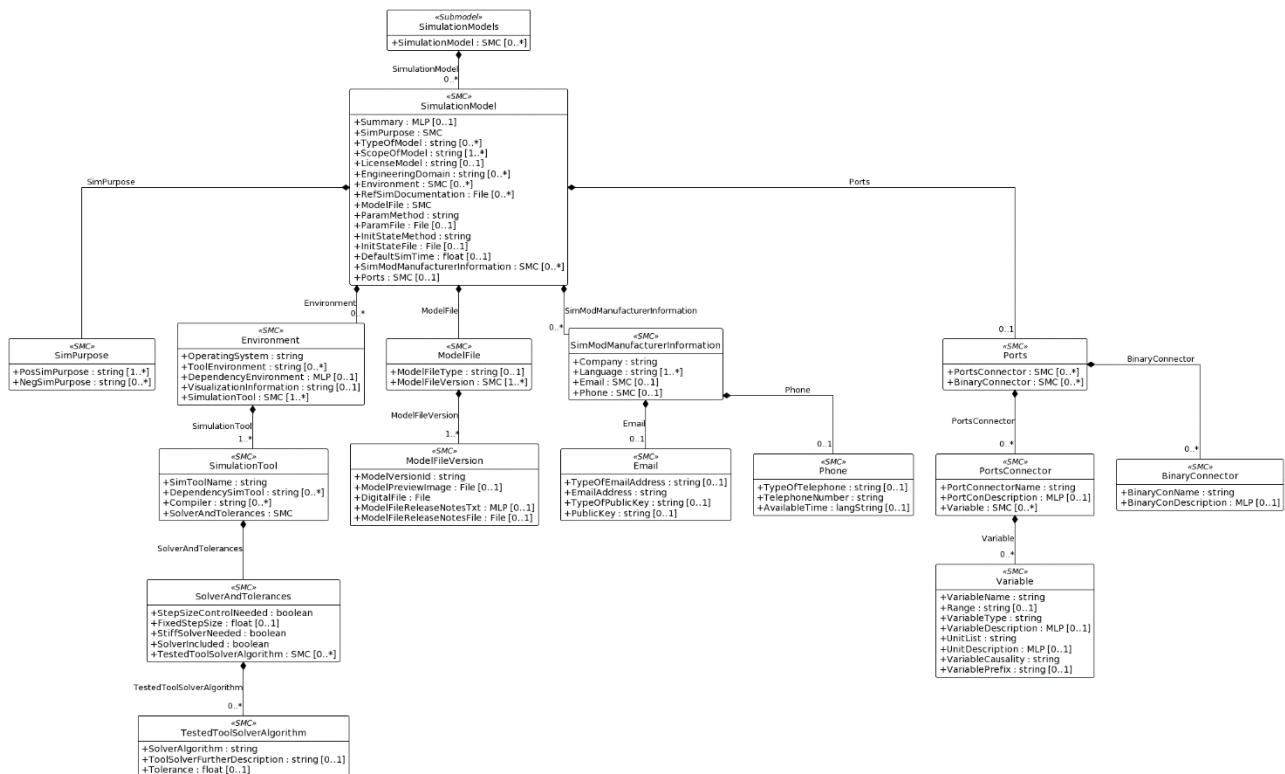


Figure 6: UML Diagram of the Submodel “Provision of Simulation Models”

2.3 Attributes of Submodel instance

For with the Submodel instance, different simulation models can be provided. The table convention is explained in Annex A.2.

Table 2: Attributes of Submodel instance

idShort:	SimulationModels		
Class:	Submodel		
semanticId:	https://admin-shell.io/idta/SimulationModels/SimulationModels/1/0		
Explanation:	The Submodel may provide one or more simulation models, a service to generate a specific model, or access to an open or specific query.		
[SME type]	semanticId = [idType]value	[valueType]	card.
idShort	Description@en	example	
[SMC] SimulationModel	[IRI] https://admin-shell.io/idta/SimulationModels/SimulationModel/1/0 Feature collection to provide or request simulation models. Models can be described by objective and content.	n/a	0..*

2.4 SubmodelElements of SimulationModel

Basic structure to describe simulation models.

Table 3: Submodel elements of SimulationModel

idShort:	SimulationModel		
Class:	SubmodelElementCollection		
semanticId:	[IRI] https://admin-shell.io/idta/SimulationModels/SimulationModel/1/0		
Parent:	SimulationModels		
Explanation:	Feature collection to provide or request simulation models. Models can be described by objective and content.		
[SME type]	semanticId = [idType]value	[valueType]	card.
idShort	Description@en	example	
[MLP] Summary	[IRI] https://admin-shell.io/idta/SimulationModels/Summary/1/0 Summary of the contents of the simulation model in text form.	[langString]	0..1
[SMC] SimPurpose	[IRI] https://admin-shell.io/idta/SimulationModels/SimPurpose/1/0 This characteristic describes the simulation purpose or suitability for different simulation goals.	n/a	1
[Property] TypeOfModel	[IRI] https://admin-shell.io/idta/SimulationModels/TypeOfModel/1/0 List of modeling approaches used for the model.	[string]	0..*
[Property] ScopeOfModel	[IRI] https://admin-shell.io/idta/SimulationModels/ScopeOfModel/1/0 List of basic physical characteristics which are represented by the model.	[string]	1..*
[Property] LicenseModel	[IRI] https://admin-shell.io/idta/SimulationModels/LicenseModel/1/0 If a simulation model usage will be charged and how it will be charged.	[string]	0..1
[Property] EngineeringDomain	[IRI] https://admin-shell.io/idta/SimulationModels/EngineeringDomain/1/0 List of engineering disciplines supported or mapped with the model.	[string]	0..*

[SMC] Environment	[IRI] https://admin-shell.io/idta/SimulationModels/Environment/1/0 Information about prerequisite environments or dependencies of underlying components on the target system.	n/a	0..*
[File] RefSimDocumentation	[IRI] https://admin-shell.io/idta/SimulationModels/RefSimDocumentation/1/0 Simulation Documentation Documentation of example simulations of the model can be supplied. This includes a solver setup and sample circuit and sample results. e.g. zip file, PDF, html, ... -	[File]	0..*
[SMC] ModelFile	[IRI] https://admin-shell.io/idta/SimulationModels/ModelFile/1/0 Providing versions of the simulation model and with characteristics to distinguish them.	n/a	1
[Property] ParamMethod	[IRI] https://admin-shell.io/idta/SimulationModels/ParamMethod/1/0 Indicates whether the model must be parameterized and if so, which method is required.	[string]	1
[File] ParamFile	[IRI] https://admin-shell.io/idta/SimulationModels/ParamFile/1/0 File for parameterization of the model. As parameter file or parameter documentation (e.g. pdf).	[File]	0..1
[Property] InitStateMethod	[IRI] https://admin-shell.io/idta/SimulationModels/InitStateMethod/1/0 Describes the state variables of the simulation model that must be initialized to start the simulation. For initial value problems, these quantities describe the system state at the start of the simulation. In this case, the system is in a state of equilibrium. Alternatively, a simulation model may include a method to determine consistent initial values at this step, e.g., at an operating point.	[string]	1
[File] InitStateFile	[IRI] https://admin-shell.io/idta/SimulationModels/InitStateFile/1/0 File for parameterizing the initial states of the model. As parameter file or parameter documentation (e.g. pdf).	[File]	0..1
[Property] DefaultSimTime	[IRI] https://admin-shell.io/idta/SimulationModels/DefaultSimTime/1/0 Predefined simulation period in seconds.	[float]	0..1
[SMC] SimModManufacturerInformation	[IRI] https://admin-shell.io/idta/SimulationModels/SimModManufacturerInformation/1/0 Provide access to simulation support service provided by the distributor via mail or phone.	n/a	0..*

[SMC] Ports	[IRI] https://admin-shell.io/idta/SimulationModels/Ports/1/0 Interfaces of the model. This includes inputs, outputs as well as acausal connections (e.g. mechanical connections). In addition, it is specified here whether the model provides binary interfaces (e.g. for visualization).	n/a	0..1
----------------	--	-----	------

2.5 SubmodelElements of SimPurpose

Table 4: Submodel elements of SimPurpose

idShort:	SimPurpose		
Class:	SubmodelElementCollection		
semanticId:	[IRI] https://admin-shell.io/idta/SimulationModels/SimPurpose/1/0		
Parent:	SimulationModel		
Explanation:	This characteristic describes the simulation purpose or suitability for different simulation goals.		
[SME type]	semanticId = [idType]value	[valueType]	card.
idShort	Description@en	example	
[Property] PosSimPurpose	[IRI] https://admin-shell.io/idta/SimulationModels/PosSimPurpose/1/0 List of simulation purposes for which the model is intended.	[string]	1..*
[Property] NegSimPurpose	[IRI] https://admin-shell.io/idta/SimulationModels/NegSimPurpose/1/0 List of simulation purposes for which the model is explicitly not suitable.	[string]	0..*

2.6 SubmodelElements of Environment

Table 5: Submodel elements of Environment

idShort:	Environment		
Class:	SubmodelElementCollection		
semanticId:	[IRI] https://admin-shell.io/idta/SimulationModels/Environment/1/0		
Parent:	SimulationModel		
Explanation:	Information about prerequisite environments or dependencies of underlying components on the target system.		
[SME type]	semanticId = [idType]value	[valueType]	card.
idShort	Description @en	example	
[Property] OperatingSystem	[IRI] https://admin-shell.io/idta/SimulationModels/OperatingSystem/1/0 Name of the operating system including version and architecture (e.g. Windows 10 64bit).	[string]	1
[Property] ToolEnvironment	[IRI] https://admin-shell.io/idta/SimulationModels/ToolEnvironment/1/0 List with required simulation tools, interpreters, model libraries or runtime libraries. In each case the exact designation of the software producer is given as free text.	[string]	0..*
[MLP] DependencyEnvironment	[IRI] https://admin-shell.io/idta/SimulationModels/DependencyEnvironment/1/0 Description of dependencies to associated hardware and software.	[langString]	0..1
[Property] VisualizationInformation	[IRI] https://admin-shell.io/idta/SimulationModels/VisualizationInformation/1/0 Ability to use a visualization. This can be integrated in a model or the model offers capabilities for connection. The connection can be described in more detail under Ports.	[string]	0..1
[SMC] SimulationTool	[IRI] https://admin-shell.io/idta/SimulationModels/SimulationTool/1/0 Properties of the model with regarding to concrete simulation tools.	n/a	1..*

2.7 SubmodelElements of SimulationTool

Table 6: Submodel elements of SimulationTool

idShort:	SimulationTool		
Class:	SubmodelElementCollection		
semanticId:	[IRI] https://admin-shell.io/idta/SimulationModels/simulationTool/1/0		
Parent:	Environment		
Explanation:	Properties of the model with regarding to concrete simulation tools.		
[SME type]	semanticId = [idType]value	[valueType]	card.
idShort	Description @en	example	
[Property] SimToolName	[IRI] https://admin-shell.io/idta/SimulationModels/SimToolName/1/0 Name of the simulation tool including version.	[string]	1
[Property] DependencySimTool	[IRI] https://admin-shell.io/idta/SimulationModels/DependencySimTool/1/0 Dependencies of Simulation Tools.	[string]	0..*
[Property] Compiler	[IRI] https://admin-shell.io/idta/SimulationModels/compiler/1/0 Name of necessary compiler including version.	[string]	0..*
[SMC] SolverAndTolerances	[IRI] https://admin-shell.io/idta/SimulationModels/SolverAndTolerances/1/0 Useful settings of the simulation environment. Includes e.g. solver settings.	n/a	1

2.8 SubmodelElements of SolverAndTolerances

Table 7: Submodel elements of SolverAndTolerances

idShort:	SolverAndTolerances		
Class:	SubmodelElementCollection		
semanticId:	[IRI] https://admin-shell.io/idta/SimulationModels/SolverAndTolerances/1/0		
Parent:	SimulationTool		
Explanation:	Useful settings of the simulation environment. Includes e.g. solver settings.		
[SME type]	semanticId = [idType]value	[valueType]	card.
idShort	Description @en	example	
[Property] StepSizeControl Needed	[IRI] https://admin-shell.io/idta/SimulationModels/StepSizeControlNeeded/1/0 Solver with step size control recommended.	[boolean]	1
[Property] FixedStepSize	[IRI] https://admin-shell.io/idta/SimulationModels/FixedStepSize/1/0 Fixed integration step size, if there is no adaptive step size.	[float]	0..1
[Property] StiffSolverNeede d	[IRI] https://admin-shell.io/idta/SimulationModels/StiffSolverNeeded/1/0 Stiff solver needed.	[boolean]	1
[Property] SolverIncluded	[IRI] https://admin-shell.io/idta/SimulationModels/SolverIncluded/1/0 Solver is integrated in the model (e.g. FMU for co-simulation).	[boolean]	1
[SMC] TestedToolSolve rAlgorithm	[IRI] https://admin-shell.io/idta/SimulationModels/TestedToolSolverAlgorithm/1/0 List of validated tool-solver combinations.	n/a	0..*

2.9 SubmodelElements of TestedToolSolverAlgorithm

Table 8: Submodel elements of TestedToolSolverAlgorithm

idShort:	TestedToolSolverAlgorithm		
Class:	SubmodelElementCollection		
semanticId:	[IRI] https://admin-shell.io/idta/SimulationModels/TestedToolSolverAlgorithm/1/0		
Parent:	SolverAndTolerances		
Explanation:	List of validated tool-solver combinations.		
[SME type]	semanticId = [idType]value	[valueType]	card.
idShort	Description @en	example	
[Property] SolverAlgorithm	[IRI] https://admin-shell.io/idta/SimulationModels/SolverAlgorithm/1/0 validated solver.	[string]	1
[Property] ToolSolverFurtherDescription	[IRI] https://admin-shell.io/idta/SimulationModels/ToolSolverFurtherDescription/1/0 Further tool- and solver-specific information.	[string]	0..1
[Property] Tolerance	[IRI] https://admin-shell.io/idta/SimulationModels/tolerance/1/0 (relative) tolerance for the adaptive step size.	[float]	0..1

2.10 SubmodelElements of ModelFile

Table 9: Submodel elements of ModelFile

idShort:	ModelFile		
Class:	SubmodelElementCollection		
semanticId:	[IRI] https://admin-shell.io/idta/SimulationModels/ModelFile/1/0		
Parent:	SimulationModel		
Explanation:	Providing versions of the simulation model and with characteristics to distinguish them.		
[SME type]	semanticId = [idType]value	[valueType]	card.
idShort	Description @en	example	
[Property] ModelFileType	[IRI] https://admin-shell.io/idta/SimulationModels/ModelFileType/1/0 Designation of the exchange format of the model. E.G.: FMI 1.0, Co-Simulation, Platform / Source - Code. FMI 2.0.2, Model Exchange, Source - Code, S-function, Version 2, 64bit, mex - Format / or C-Code, Modelica 3, encoded, VHDL	[string]	0..1
[SMC] ModelFileVersion	[IRI] https://admin-shell.io/idta/SimulationModels/ModelFileVersion/1/0 Provision of a version of the simulation model with information to distinguish the versions. The versions are primarily intended for bug fixes without content changes.	n/a	1..*

2.11 SubmodelElements of ModelFileVersion

Table 10: Submodel elements of ModelFileVersion

idShort:	ModelFileVersion		
Class:	SubmodelElementCollection		
semanticId:	[IRI] https://admin-shell.io/idta/SimulationModels/ModelFileVersion/1/0		
Parent:	ModelFile		
Explanation:	Provision of a version of the simulation model with information to distinguish the versions. The versions are primarily intended for bug fixes without content changes.		
[SME type]	semanticId = [idType]value	[valueType]	card.
idShort	Description @en	example	
[Property] ModelVersionId	[IRI] https://admin-shell.io/idta/SimulationModels/ModelVersionId/1/0 Version number of the model from the vendor.	[string]	1
[File] ModelPreviewImage	[IRI] https://admin-shell.io/idta/SimulationModels/ModelPreviewImage/1/0 Image file to represent the model in user interfaces, e.g. in a search.	[File]	0..1
[File] DigitalFile	[IRI] https://admin-shell.io/idta/SimulationModels/DigitalFile/1/0 Deployment of the model file.	[File]	1
[MLP] ModelFileReleaseNotesTxt	[IRI] https://admin-shell.io/idta/SimulationModels/ModelFileReleaseNotesTxt/1/0 contains information about this release	[langString]	0..1
[File] ModelFileReleaseNotesFile	[IRI] https://admin-shell.io/idta/SimulationModels/ModelFileReleaseNotesFile/1/0 release notes link or file	[File]	0..1

2.12 SubmodelElements of SimModManufacturerInformation

Table 11: Submodel elements of SimModManufacturerInformation

idShort:	SimModManufacturerInformation		
Class:	SubmodelElementCollection		
semanticId:	[IRI] https://admin-shell.io/idta/SimulationModels/SimModManufacturerInformation/1/0		
Parent:	SimulationModel		
Explanation:	Provide access to simulation support service provided by the distributor via mail or phone.		
[SME type]	semanticId = [idType]value	[valueType]	card.
idShort	Description@en	example	
[Property] Company	[IRDI] 0173-1#02-AAW001#001 name of the company	[string]	1
[Property] Language	[IRDI] 0173-1#02-AAO895#003 available language	[string]	1..*
[SMC] Email	[IRDI] 0173-1#02-AAQ836#005 E-mail address and encryption method	n/a	0..1
[SMC] Phone	[IRI] https://admin-shell.io/zvei/nameplate/1/0/ContactInformations/ContactInformation/Phone Phone number including type	n/a	0..1

2.13 SubmodelElements of Email

Table 12: Submodel elements of Email

idShort:	Email Note: according to Submodel "Contact Information"		
Class:	SubmodelElementCollection		
semanticId:	[IRDI] 0173-1#02-AAQ836#005		
Parent:	SimModManufacturerInformation		
Explanation:	E-mail address and encryption method		
[SME type]	semanticId = [idType]value	[valueType]	card.
idShort	Description @en	example	
[Property] TypeOfEmailAd dress	[IRDI] 0173-1#02-AAO199#003 characterization of an e-mail address according to its location or usage	[string]	0..1
[Property] EmailAddress	[IRDI] 0173-1#02-AAO198#002 electronic mail address of a business partner	[string]	1
[Property] TypeOfPublicKe y	[IRDI] 0173-1#02-AAO201#002 characterization of a public key according to its encryption process	[string]	0..1
[Property] PublicKey	[IRDI] 0173-1#02-AAO200#002 public part of an unsymmetrical key pair to sign or encrypt text or messages	[string]	0..1

2.14 SubmodelElements of Phone

Table 13: Submodel elements of Phone

idShort:	Phone Note: according to Submodel "Contact Information"		
Class:	SubmodelElementCollection		
semanticId:	[IRI] https://admin-shell.io/zvei/nameplate/1/0/ContactInformations/ContactInformation/Phone		
Parent:	SimModManufacturerInformation		
Explanation:	Phone number including type		
[SME type]	semanticId = [idType]value	[valueType]	card.
idShort	Description@en	example	
[Property] TypeOfTelephone	[IRDI] 0173-1#02-AAO137#003 characterization of a telephone according to its location or usage	[string]	0..1
[Property] TelephoneNumber	[IRDI] 0173-1#02-AAO136#002 complete telephone number to be called to reach a business partner	[string]	1
[MLP] AvailableTime	[IRI] https://admin-shell.io/zvei/nameplate/1/0/ContactInformations/ContactInformation/AvailableTime Specification of the available time window	[langString]	0..1

2.15 SubmodelElements of Ports

Table 14: Submodel elements of Ports

idShort:	Ports		
Class:	SubmodelElementCollection		
semanticId:	[IRI] https://admin-shell.io/idta/SimulationModels/Ports/1/0		
Parent:	SimulationModel		
Explanation:	Interfaces of the model. This includes inputs, outputs as well as acausal connections (e.g. mechanical connections). In addition, it is specified here whether the model provides binary interfaces (e.g. for visualization).		
[SME type]	semanticId = [idType]value	[valueType]	card.
idShort	Description @en	example	
[SMC] PortsConnector	[IRI] https://admin-shell.io/idta/SimulationModels/PortsConnector/1/0 List of ports of the model. These include a name, a description, a list of variables, and a list of ports.	n/a	0..*
[SMC] BinaryConnector	[IRI] https://admin-shell.io/idta/SimulationModels/BinaryConnector/1/0 Binary interfaces (binaryType) based on the FMI 3.0 standard (https://fmi-standard.org/docs/3.0-dev/#definition-of-types). At this point the name (e.g. "Binary interface visualization") and the description (e.g. "Interface for binary transfer of visualization information") are specified.	n/a	0..*

2.16 SubmodelElements of PortsConnector

Table 15: Submodel elements of PortsConnector

idShort:	PortsConnector		
Class:	SubmodelElementCollection		
semanticId:	[IRI] https://admin-shell.io/idta/SimulationModels/PortsConnector/1/0		
Parent:	Ports		
Explanation:	List of ports of the model. These include a name, a description, a list of variables, and a list of ports.		
[SME type]	semanticId = [idType]value	[valueType]	card.
idShort	Description@en	example	
[Property] PortConnectorName	[IRI] https://admin-shell.io/idta/SimulationModels/PortConnectorName/1/0 Name of the Connector Port.	[string]	1
[MLP] PortConDescription	[IRI] https://admin-shell.io/idta/SimulationModels/PortConDescription/1/0 Description of the Connector Port.	[langString]	0..1
[SMC] Variable	[IRI] https://admin-shell.io/idta/SimulationModels/Variable/1/0 List of variables of the port.	n/a	0..*

2.17 SubmodelElements of Variable

Table 16: Submodel elements of Vaiaable

idShort:	Variable		
Class:	SubmodelElementCollection		
semanticId:	[IRI] https://admin-shell.io/idta/SimulationModels/Variable/1/0		
Parent:	PortsConnector		
Explanation:	-		
[SME type]	semanticId = [idType]value	[valueType]	card.
idShort	Description @en	example	
[Property] VariableName	[IRI] https://admin-shell.io/idta/SimulationModels/VariableName/1/0 Name of the variable.	[string]	1
[Property] Range	[IRI] https://admin-shell.io/idta/SimulationModels/Range/1/0 Range of values for the variable (e.g. [min, max], [min, max[,]min, max[,]min, max[, {val1, val2, ...}]). -	[string]	0..1
[Property] VariableType	[IRI] https://admin-shell.io/idta/SimulationModels/VariableType/1/0 Type of the variable (e.g. Real, Integer, Boolean, String or Enum).	[string]	1
[MLP] VariableDescription	[IRI] https://admin-shell.io/idta/SimulationModels/VariableDescription/1/0 Description of the variable.	[langString]	0..1
[Property] UnitList	[IRI] https://admin-shell.io/idta/SimulationModels/UnitList/1/0 The most common units can be selected here. ... If "others" is selected, a free text can be entered.	[string]	1
[MLP] UnitDescription	[IRI] https://admin-shell.io/idta/SimulationModels/UnitDescription/1/0 Text field for missing units of the list	[langString]	0..1
[Property] VariableCausality	[IRI] https://admin-shell.io/idta/SimulationModels/VariableCausality/1/0 The causality of the variable: input to inputs, output to outputs, acausal connections (e.g. mechanical connection) do not have causality.	[string]	1
[Property] VariablePrefix	[IRI] https://admin-shell.io/idta/SimulationModels/VariablePrefix/1/0 Prefix for acausal variable. Potential variables are set equal when connecting (no prefix). "flow" variables are connected according to Kirchhoff's law, i.e. the sum of the variables equals zero. The bi-directional flow of matter is described by the prefix "stream" (e.g. for enthalpy).	[string]	0..1

2.18 SubmodelElements of BinaryConnector

Table 17: Submodel elements of BinaryConnector

idShort:	BinaryConnector		
Class:	SubmodelElementCollection		
semanticId:	[IRI] https://admin-shell.io/idta/SimulationModels/BinaryConnector/1/0		
Parent:	Ports		
Explanation:	Binary interfaces (binaryType) based on the FMI 3.0 standard (https://fmi-standard.org/docs/3.0-dev/#definition-of-types). At this point the name (e.g. "Binary interface visualization") and the description (e.g. "Interface for binary transfer of visualization information") are specified.		
[SME type]	semanticId = [idType]value	[valueType]	card.
idShort	Description @en	example	
[Property] BinaryConName	[IRI] https://admin-shell.io/idta/SimulationModels/BinaryConnectorName/1/0 Binary interface name.	[string]	1
[Property] BinaryConDescription	[IRI] https://admin-shell.io/idta/SimulationModels/BinaryConDescription/1/0 Binary interface description.	[string]	0..1

2.19 Predefined values for properties

The value lists listed here for certain properties are also contained in the AASX SMT published with the specification.

The values should be mapped in a semantic dictionary in the future, like eCI@ss. Until then, the value lists can be used. Depending on the property, it is recommended in the following subchapters whether a value selection should be closed or open.

"closed" ... it is recommended to use one of the selections.

"open" ... there is a possibility to add your own definition

2.19.1 Value list for PosSimPurpose and NegSimPurpose

This list of values is recommended as an open selection. The model provider should be able to add more definitions.

Table 18: Value list for PosSimPurpose and NegSimPurpose

PreferredName@en	Description@en
Concept evaluation	Concept evaluation
Sizing	Sizing, check of requirements, check of cycle time, etc.
Energy consumption	Energy consumption, energy management, etc.
Control design	Linear model for control design
Behaviour in fault condition	Fault management, safety engineering, etc.
System dynamics	Checking the dynamic behavior of a component as part of a system of components, etc.
Virtual commissioning	Virtual commissioning, especially testing of bus communication, software, timing, etc.
Condition monitoring	Condition monitoring
Predictive maintenance	Predictive maintenance
Operator Training	Training, plant specific training, etc.
Teaching	College and academic teaching

2.19.2 Value list for TypeOfModel

This list of values is recommended as an open selection. The model provider should be able to add more definitions.

Table 19: Value list for TypeOfModel

PreferredName@en	Description@en
Linear model	Models which do not contain any nonlinear functions of time-variant variables (input-/output-/state variables), such as: Multiplication, power-, or trigonometric functions. Linear models follow the superposition principle.
Nonlinear model	Models which contain nonlinear functions of time-variant variables (input-/output-/state variables). A nonlinear model may still contain linear functions, but the presence of at least one nonlinear function is sufficient to make it nonlinear.
Data-driven model	Blackbox models, such as LUT (Look-up Table), polynomial functions, neural networks, machine learning, etc.
Lumped element model	Lumped element models represents the properties of a system, e.g. a wire and capacitor in the electrical domain or a shaft and spring in the mechanic domain, which are represented by (spatially) discretized components with discrete parameters. It should be noted, that this property does not limit the model in terms of its representation, i.e., a model with lumped elements can still be modeled as either acausal, e.g. model in network representation, or causal, e.g. state space model.
Fixed causality model	In this model, the mathematical part is described by fixed causality relations between port variables. The connections in this model are unidirectional and the ports are defined as input and output ports.
Acausal model	In the model, the mathematical descriptions are in acausal form. For the evaluation of the model in general, systems of algebraic equations must be solved numerically. The connections in these models are bidirectional, e.g. network model of a lossy electrical line. Please note, acausal filtering algorithms [FIR/IIR] do not belong to this model category.

2.19.3 Value list for ScopeOfModel

This list of values is recommended as an open selection. The model provider should be able to add more definitions.

Table 20: Value list for ScopeOfModel

PreferredName@en	Description@en
Logic and timing behaviour	State machines, models of operational sequence, ...
Geometry	Geometry based model for e.g. assessment of collision or assembly, computation of geometry-based properties like envelope, mass, inertia, reference coordinate systems, position of connecting points, ...
Kinematics	Kinematic model, computation of velocity, acceleration and jerk, degrees of freedom, range of motion, ...
Dynamics	Dynamic model, computation of conservation variables, ...
Distribution networks	Distribution networks like water, gas, power, material flow, ...
Communication networks	Communication networks
Visualization	Visualization, model GUI, ...

2.19.4 Value list for LicenseModel

This list of values is recommended as an open selection. The model provider should be able to add more definitions.

Table 21: Value list for LicenseModel

PreferredName@en
free
perpetual
subscription
volume-based

2.19.5 Value list for EngineeringDomain

This list of values is recommended as an open selection. The model provider should be able to add more definitions.

Table 22: Value list for EngineeringDomain

PreferredName@en
Hydraulic Engineering
Electrical Engineering
Pneumatic Engineering
Mechanical Engineering
Electronics Engineering
Thermal Engineering
Material Flow
Robotics
Image Processing
Data Engineering
Process Engineering
Workflow Engineering
HMI Engineering
Control Engineering

2.19.6 Value list for VisualizationInformation

This list of values is recommended as a closed selection. Only these values should be taken.

Table 23: Value list for VisualizationInformation

PreferredName@en	Description@en
separately	The usage of an external visualization (e.g. Collada) is intended. An interface description can be provided in the section "ports".
integrated	The visualization information is integrated in the model. The model can be visualized with a compatible visualization software.
none	No visualization of the model is intended.

2.19.7 Value list for ParamMethod

This list of values is recommended as an open selection. The model provider should be able to add more definitions.

Table 24: Value list for ParamMethod

PreferredName@en	Description@en
by technical data of asset	The parameterization is completely included in the technical data. These can be taken over automatically.
by technical data and user interface	The parameterization is partly included in the technical data. These can be taken over automatically.
by user interface	The model can be parameterized.
by settings file	A parameter file is available that can be read in by the simulation tool.
not necessary	The model does not require parameterization.
by documentation file	The parameterization is described in a document and can be adopted into the model.
pre-parameterized	The model is pre-parameterized. Individual values can or must still be adjusted.

2.19.8 Value list for InitStateMethod

This list of values is recommended as an open selection. The model provider should be able to add more definitions.

Table 25: Value list for InitStateMethod

PreferredName@en	Description@en
not necessary	models without states
by user interface	initial states set via UI
by setting file	a file for setting is available
set states within simulation environment	initStateCalculation triggered by simulation environment
integrated in model	sinnvolle Werte sind im Model gesetzt
by documentation file	the documentation contains the limits of the state variables and describes the dependencies due to the model that must be considered

2.19.9 Value list for VariableType

This list of values is recommended as an open selection. The model provider should be able to add more definitions.

Table 26: Value list for VariableType

PreferredName@en
Real
Integer
Boolean
String
ENUM

2.19.10 Value list for VariableCausality

This list of values is recommended as a closed selection. Only these values should be taken.

Table 27: Value list for VariableCausality

PreferredName@en
Input
Output
Acausal

2.19.1 Value list for VariablePrefix

This list of values is recommended as a closed selection. Only these values should be taken.

Table 28: Value list for VariablePrefix

PreferredName@en
Flow
Stream

3 AAS SM Modeling Scenarios

3.1 Managing multiple models

Due to its structure, the asset administration shell allows a certain freedom in the modeling of its contents. This freedom is intentional and necessary to give information modelers the freedom to adapt the structure of the AAS to the existing constructs of their own companies and thus increase the applicability of the AAS. In this sense, this section does not represent a strict modeling regulation that specifies how simulation models should be hierarchised. Rather, it is intended to provide a recommendation for the implementation that makes sense from the point of view of the working group and should ideally be used to enable a uniform structuring of the Submodel and the simulation models across several AAS.

As can be seen from the structure of the Submodel template, the Submodel "Provision of Simulation Models" offers the possibility of depositing several different simulation models as well as different versions of identical simulation models in an AAS. A recommendation on how such structures from several simulation models should be stored is presented below. The following points are addressed:

- Multiple versions of one simulation model
- Multiple different simulation models

3.1.1 Explanatory example:

The following example assumes that there are three different simulation models (Model A, Model B and Model C). Model A has versions V2.0 and V2.1, Model B only has version 8.0 and Model C has three versions V5.0, V5.1 and V5.2. The structure and chronological development are shown again in Figure 7.

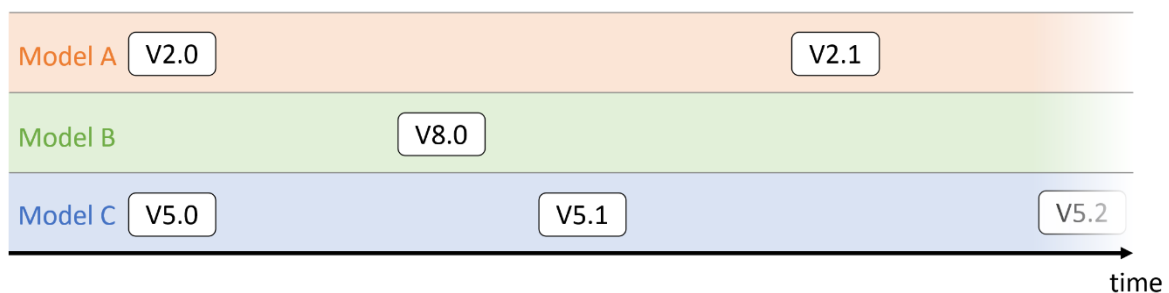


Figure 7: Structure and chronological development of the simulation models.

3.1.2 Recommended modeling:

In general, all simulation models of a component are encapsulated in a single Submodel contained in the AAS of the Component. A separate SubmodelCollection "SimulationModel" is created for each simulation model. Applied to the example above, there would be one Submodel "SimulationModels" in the administration shell of the component, which itself contains six SubmodelCollections "SimulationModel". Each of these SubmodelCollections thus represents a simulation model in a specific version. This modeling variant is shown in Figure 8. The structure of the Submodel template also allows several versions of one simulation model to be represented within a SubmodelCollection "SimulationModel" as can be seen in Figure 8 (Model C V5.0 and V5.1). However, this modeling structure is only recommended if the description of the

SubmodelCollection "Simulation Model" does not change. More detailed information can be found in the subsections "Bug Fixes" and "New Information Model".

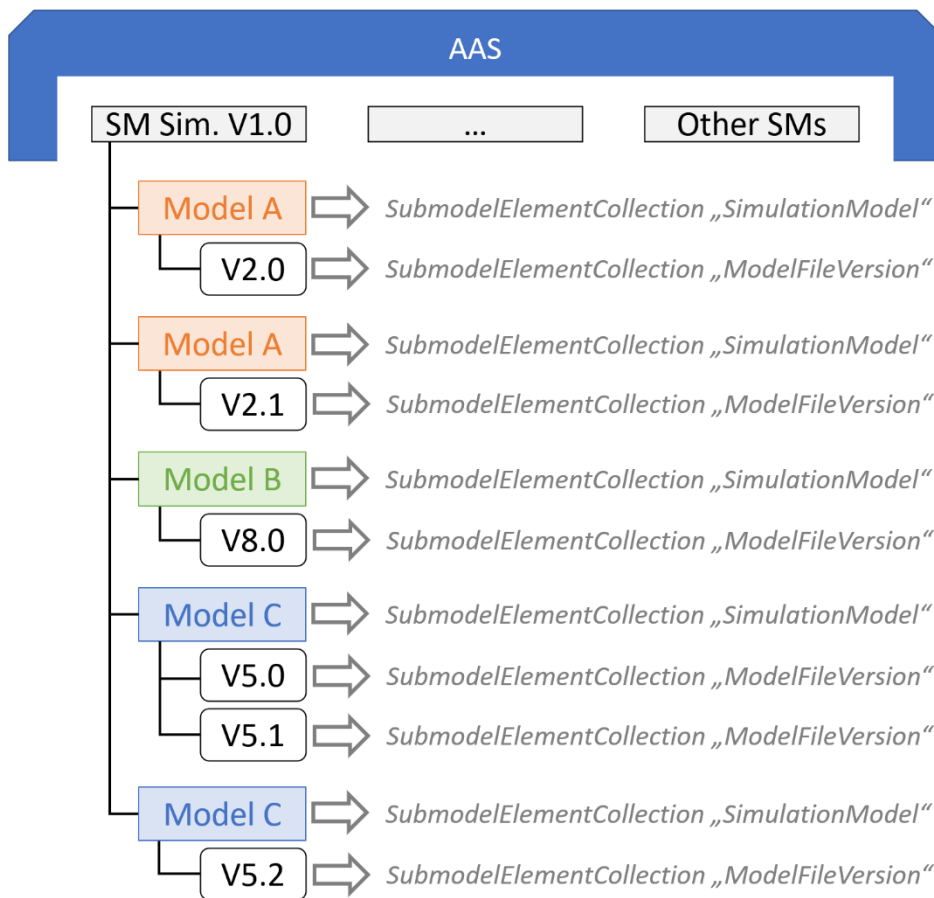


Figure 8: Modeling approaches for the Submodel “Provision of Simulation Models”

The following are recommendations for modeling simulation models with the “Provision of Simulation Models” Submodel for different situations.

3.1.3 Simulation Model Bugfix

Simulation Model C should serve as an example here. Consider the situation, that in V5.0, the pre-factor of an equation is incorrect and must be adjusted in the simulation model itself. The change of the original (V5.0) and corrected model (V5.1) differ only in the referenced file "DigitalFile", representing the new simulation model. For this reason, the corrected version V5.1 can be added to the same SubmodelCollection "Simulation Model" as the previous version 5.0. For this purpose, an additional SubmodelCollection "modelFileVersion" is added to the SubmodelCollection "modelFile", containing the corrected version of the Model C, a different version number, etc.. The structure is shown in Figure 8 and below as a hierarchical structure.

- SimulationModels
 - o SimulationModel „C“
 - ModelFile „C“
 - ModelFileVersion „1“
 - o ModelFileVerComment „First Version“
 - o ModelFileVerId „V5.0“
 - **ModelFileVersion „2“**
 - o **ModelFileVerComment „Bugfix in prefactor of equation xx“**
 - o **ModelFileVerId „V5.1“**

- Summary „Model for simulating control behavior.“
- ...
- Ports ... all versions must match the descriptions

However, if the Bugfix results in a different representation of the simulation model within the SubmodelCollection “SimulationModel”, the usage of a new SubmodelCollection “SimulationModel” is necessary. This scenario is described in the following subsection.

3.1.4 New Simulation Model

To correctly represent simulation models with different properties, a new SubmodelCollection “SimulationModel” need to be added to the Submodel, regardless if it is a completely new simulation model or a new version of an already existing one. For example, Model C V5.2 contains an additional Port to read an extra Variable value. Since this property is not visible in the AAS description of V5.0 and V5.1, a new SubmodelCollection “SimulationModel” must be created for Model C V5.2. Below the structure of the scenario is depicted containing also a completely different Model A.

- SimulationModels
 - o SimulationModel „C“
 - ModelFile „C“
 - ModelFileVersion „1“
 - o ModelFileVerComment „First Version“
 - o ModelFileVerId „V5.0“
 - ModelFileVersion „2“
 - o ModelFileVerComment „1. Bugfix in prefactor in equation xx“
 - o ModelFileVerId „V5.1“
 - Summary „Model for simulating control behavior. Modeltype 1.1“
 - ...
 - Ports ... all versions must match the descriptions
 - o **SimulationModel “C”**
 - **ModelFile „C“**
 - **ModelFileVersion „3“**
 - o **ModelFileVerComment „Adding a port to read variable xx“**
 - o **ModelFileVerId „V5.2“**
 - **Summary „Model for simulating control behavior. Modeltype 1.2“**
 - ...
 - **Ports ... containing a different description than V5.0 and V5.1**
 - o **SimulationModel “A”**
 - **ModelFile „A“**
 - **ModelFileVersion „1“**
 - o **ModelFileVerComment „First Version“**
 - o **ModelFileVerId „V2.0“**
 - **Summary „Model for virtual commissioning. Modeltyp 2.1“**
 - ...
 - **Ports ...**

3.2 Model requests to component supplier

As described in the requirement 041 and 042 of Table 1: Requirements to the Submodel "Provision of Simulation Models", in addition to providing simulation models, the Submodel can also support requesting simulation models for a component from the supplier. Two examples of use are shown below.

3.2.1 Offering explicit models

A component provider can create a model for its component or solution on request. He provides this only on request, because the provision may not work completely automated or is too costly.

The partial model can be used as follows.

- The Submodel “Provision of Simulation Models” is added to the AAS of the component.
- No simulation file is added. The version is left empty. A preview image representing the model can be optionally added.
- The contact information can be used to allow a reply by mail.
- The properties describing the simulation model can be used to present the features of the model as transparently as possible.

Below is a schematic example:

- SimulationModels
 - SimulationModel „01“
 - ModelFile „01“
 - ModelFileVersion „01“
 - ModelFileVerId „“
 - DigitalFile <empty>
 - PreviewFile <optional>
 - Summary „Model for simulating control behavior. Please asked for it if needed or useful. “
 - ...
 - SimModManufacturerInformation
 - Company “Supplier xy”
 - Language “en”, “de”
 - Email
 - EmailAddress “SimulationSupport@Supplier-xy.com”
 - ...
 - ...
 - SimPurpose ...
 - ScopeOfModel ...
 - Environment
 - ...

3.2.2 Query of individual model

A component supplier wants to receive requests for simulation models of its components from users. The Submodel can help to ensure that the requests for simulation models are made accurately. Tools, like a simulation environment, but also e.g. a PLM system, can support the organization of needed things, like simulation models, based on this.

The partial model can be used as follows.

- The Submodel “Provision of Simulation Models” is added to the AAS of the component.
- No simulation file is added. The version is left empty.
- The contact information can be used to allow a reply by mail. Other Submodels of the components AAS, like the “Digital Nameplate for Industrial Equipment”, can be used to reply to useful information to clarify.
- The properties that describe the simulation model can be used to define an application range the supplier is able to offer.

- Or the tools that organize the purchase of models for the potentially useable components and solutions use the properties as a form. These can then pass on the requirements to the suppliers in a standardized way.

Below is a schematic example:

- SimulationModels
 - SimulationModel „01“
 - ModelFile „01“
 - ModelFileVersion „01“
 - ModelFileVerId „“
 - DigitalFile <empty>
 - PreviewFile <optional>
 - Summary „Diverse Simulationsmodel can be available by request. Please use the properties to define your need. “
 - SimModManufacturerInformation
 - Company “Supplier xy”
 - Language “en”, “de”
 - Email
 - EmailAddress “SimulationSupport@Supplier-xy.com”
 - ...
 - ...
 - SimPurpose ...
 - ScopeOfModel ...
 - Environment
 - SimulationTool
 - ...

3.3 Modeling Supported Environments

Some simulation models may run on various environments. A FMU can contain a model for Windows, Linux and other platforms, and also for various architectures like x86, x64. A simulation model written in Open Modelica can run on various environments and various simulation tools.

As creator of a simulation model, the provision of these information within the AAS is necessary, so users or programs can check whether the model runs on their simulation environment or not. The following figure shows two supported operating systems Windows and Ubuntu with their supported simulation Tools.

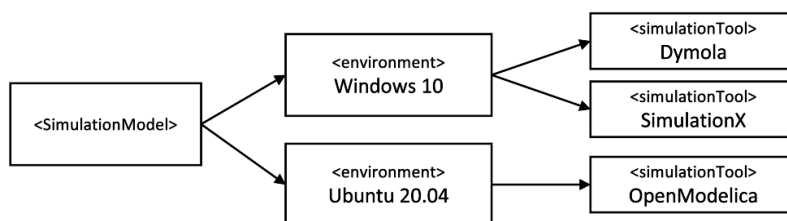


Figure 9: Various environments, represented by operating systems and simulation tools

Sometimes you will also need to provide more information about the dependencies of a simulation tool itself, e. g. for Modelica based models you want to provide information about the used compiler and the tested solver algorithms. The next figure shows how to model this.

This library is designed to work with every modelica tool which supports the Modelica Standard Library 3.2.3

Supported operating systems

This library was developed and tested with Windows 10 64 bit only.

Supported Modelica Environments

However different Modelica compilers may provide different experience.

Below you find a list of the tools with their modelica compiler as well as c compilers this library has been tested with.

Tool	Version	Compiler	Solver	Tolerance	Support
Dymola	2022	Visual Studio 2019 / MinGW-w64 10.2	Dassl	1e-6	yes
SimulationX	4.2.4	Visual Studio 2019 / GNU C/C++ Compiler (tdm64-1) 5.1.0	CVODE	1e-6	yes
Open Modelica	1.18	MinGW-w64 10.2	Dassl	1e-6	no

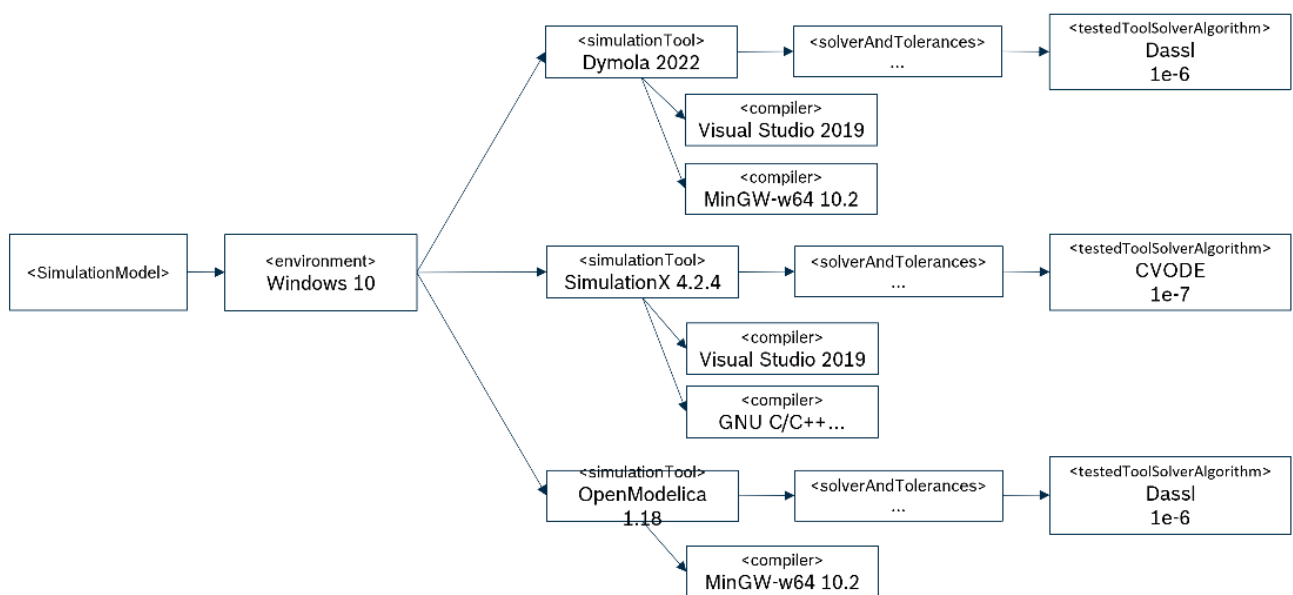


Figure 10: Simulation tools with compiler and solvers

The example above can be modeled like this:

- SimulationModel „01”
 - Environment#1
 - OperatingSystem „Windows 10 x64”
 - SimulationTool#1
 - SimToolName „Dymola 2022”
 - DependencySimTool „Modelica Standard Library 3.2.3”
 - Compiler#1 „Visual Studio 2019”
 - Compiler#2 „MinGW-w64 10.2”
 - SolverAndTolerances
 - TestedToolSolverAlgorithm
 - SolverAlgorithm „Dassl”
 - Tolerance: „1e-6”
 - SimulationTool#2
 - SimToolName „SimulationX 4.2.4”
 - DependencySimTool „Modelica Standard Library 3.2.3”
 - Compiler#1 „Visual Studio 2019”

- Compiler#2 „GNU C/C++ Compiler (tdm64-1) 5.1.0”
- SolverAndTolerances
 - TestedToolSolverAlgorithm
 - SolverAlgorithm „CVODE”
 - Tolerance „1e-7”
- SimulationTool#3
 - SimToolName “Open Modelica 1.18”
 - DependencySimTool “Modelica Standard Library 3.2.3 “
 - Compiler#1 “MinG-w64 10.2”
 - SolverAndTolerances
 - TestedToolSolverAlgorithm
 - SolverAlgorithm „Dassl”
 - Tolerance „1e-6”

Annex A. Explanations on used table formats

1. General

The used tables in this document try to outline information as concise as possible. They do not convey all information on Submodels and SubmodelElements. For this purpose, the definitive definitions are given by a separate file in form of an AASX file of the Submodel template and its elements.

2. Tables on Submodels and SubmodelElements

For clarity and brevity, a set of rules is used for the tables for describing Submodels and SubmodelElements.

- The tables follow in principle the same conventions as in [5].
- The table heads abbreviate 'cardinality' with 'card'.
- The tables often place two informations in different rows of the same table cell. In this case, the first information is marked out by sharp brackets [] from the second information. A special case are the semanticIds, which are marked out by the format: (type)(local)[idType]value.
- The types of SubmodelElements are abbreviated:

SME type	SubmodelElement type
Property	Property
MLP	MultiLanguageProperty
Range	Range
File	File
Blob	Blob
Ref	ReferenceElement
Rel	RelationshipElement
SMC	SubmodelElementCollection

- If an idShort ends with '{00}', this indicates a suffix of the respective length (here: 2) of decimal digits, in order to make the idShort unique. A different idShort might be chosen, as long as it is unique in the parent's context.
- The Keys of semanticId in the main section feature only idType and value, such as:
[IRI] <https://admin-shell.io/vdi/2770/1/0/DocumentId/Id>. The attributes "type" and "local" (typically "ConceptDescription" and "(local)" or "GlobalReference" and "(no-local)") need to be set accordingly; see [6].
- If a table does not contain a column with "parent" heading, all represented attributes share the same parent. This parent is denoted in the head of the table.
- Multi-language strings are represented by the text value, followed by '@'-character and the ISO 639 language code: example@EN.
- The [valueType] is only given for Properties.

Bibliography

- [1] "Recommendations for implementing the strategic initiative INDUSTRIE 4.0", acatech, April 2013.[Online]. Available: <https://www.acatech.de/Publikation/recommendations-for-implementing-the-strategic-initiative-industrie-4-0-final-report-of-the-industrie-4-0-working-group/>
- [2] "Implementation Strategy Industrie 4.0: Report on the results of the Industrie 4.0 Platform"; BITKOM e.V. / VDMA e.V., /ZVEI e.V., April 2015. [Online]. Available: <https://www.bitkom.org/noindex/Publikationen/2016/Sonstiges/Implementation-Strategy-Industrie-40/2016-01-Implementation-Strategy-Industrie40.pdf>
- [3] "The Structure of the Administration Shell: TRILATERAL PERSPECTIVES from France, Italy and Germany", March 2018, [Online]. Available: <https://www.plattform-i40.de/I40/Redaktion/EN/Downloads/Publikation/hm-2018-trilaterale-coop.html>
- [4] "Beispiele zur Verwaltungsschale der Industrie 4.0-Komponente – Basisteil (German)"; ZVEI e.V., Whitepaper, November 2016. [Online]. Available: <https://www.zvei.org/presse-medien/publikationen/beispiele-zur-verwaltungsschale-der-industrie-40-komponente-basisteil/>
- [5] "Verwaltungsschale in der Praxis. Wie definiere ich Teilmodelle, beispielhafte Teilmodelle und Interaktion zwischen Verwaltungsschalen (in German)", Version 1.0, April 2019, Plattform Industrie 4.0 in Kooperation mit VDE GMA Fachausschuss 7.20, Federal Ministry for Economic Affairs and Energy (BMWi), Available: <https://www.plattform-i40.de/PI40/Redaktion/DE/Downloads/Publikation/2019-verwaltungsschale-in-der-praxis.html>
- [6] "Details of the Asset Administration Shell; Part 1 - The exchange of information between partners in the value chain of Industrie 4.0 (Version 3.0RC01)", November 2020, [Online]. Available: https://www.plattform-i40.de/IP/Redaktion/EN/Downloads/Publikation/Details_of_the_Asset_Administration_Shell_Part1_V3.html
- [7] "Eine systematische Bewertung der Qualität von Simulationsmodellen für die Automatisierungstechnik – Identifikation und Clustering von Qualitätskriterien". Barth, M.; Kübler, K.; Heinzerling, T.; Rosen, R.; Jäkel, J. : In: Automation 2020: VDI Verlag, S. 499–516.
- [8] "Modelica Association Project" ; Available: <https://modelica.org/projects>
- [9] "Specification of the Functional Mock-Up Interface (FMI)", [Online] Available: [modelica/fmi-standard: Specification of the Functional Mock-Up Interface \(FMI\) \(github.com\)](https://modelica.org/fmi-standard:Specification-of-the-Functional-Mock-Up-Interface-(FMI)(github.com))
- [10] "MIC: Model Identity Card" ; [Online] Available: <https://mic.irt-systemx.fr/mic>
- [11] "AutomotionML" ; [Online] Available: <https://www.automationml.org/>
- [12] "AutomationML Application Recommendation Asset Administration Shell Representation" ; November 2019, [Online] Available: https://www.automationml.org/wp-content/uploads/2021/06/Asset-Administration-Shell-Representation-V1_0_0.zip

- [13] "Interoperabilität hoch zwei für Simulationsmodelle"; November 2020, [Online] Available: <https://www.plattform-i40.de/IP/Redaktion/DE/Downloads/Publikation/Interoperabilit%C3%A4t-hoch-zwei.html>

- [14] „Fortschreibung der Anwendungsszenarien der Plattform Industrie 4.0“, Plattform Industrie 4.0, July 2016. [Online]. Available: [Plattform Industrie 4.0 - Aspects of the Research Roadmap in Application Scenarios \(plattform-i40.de\)](#)

www.industrialdigitaltwin.org