Document Title	Adaptive Platform Machine Configuration
Document Category	Template specification (TPS)
Document UID	1112
Document Status	published
Part of AUTOSAR Standard	Adaptive Platform
Part of Standard Release	R26-11

Document Change History

Date	Release	Changed by	Description
2026-11-26	R26-11	AUTOSAR Release Management	• Harmonized “meta-requirements” to M1 models with EcuC
2025-11-27	R25-11	AUTOSAR Release Management	• Support interaction between functional clusters • Add parameters that conform to specific regular expressions • Add reference to Traceable
2024-11-27	R24-11	AUTOSAR Release Management	• Initial release of the document

Disclaimer

This work (specification and/or software implementation) and the material contained in it, as released by AUTOSAR, is for the purpose of information only. AUTOSAR and the companies that have contributed to it shall not be liable for any use of the work.

The material contained in this work is protected by copyright and other types of intellectual property rights. The commercial exploitation of the material contained in this work requires a license to such intellectual property rights.

This work may be utilized or reproduced without any modification, in any form or by any means, for informational purposes only. For any other purpose, no part of the work may be utilized or reproduced, in any form or by any means, without permission in writing from the publisher.

The work has been developed for automotive applications only. It has neither been developed, nor tested for non-automotive applications.

The word AUTOSAR and the AUTOSAR logo are registered trademarks.

Table of Contents

1 Introduction 7
1.1 Document Conventions . . . 7
2 Conceptual Aspects 9
2.1 Design and Target Configuration Models . . . 9
2.1.1 Design Model . . . 9
2.1.2 Target Configuration Models . . . 10
2.2 Overview . . . 10
2.3 Benefits of using an M1 model-based configuration approach . . . 12
2.4 Modeling Strategies for deriving M1 Model Content . . . 12
2.4.1 Only Design exists . . . 12
2.4.2 Fully derive Design Model to Configuration Model . . . 13
2.4.3 Partially derive Design Model to Configuration Model . . . 13
2.4.4 Only Configuration exists . . . 14
2.5 Vendor Specific Functional Cluster Definition . . . 15
2.6 Life Cycle definition . . . 15
3 Overview 17
3.1 Abstract Structure . . . 17
3.2 Definition Side . . . 19
3.2.1 Definition Elements . . . 20
3.2.2 Grouping of Configuration Elements . . . 23
3.2.3 Emulation of Inheritance . . . 25
3.2.4 Parameter . . . 26
3.2.5 Reference . . . 26
3.3 Reference to Traceable Elements . . . 27
3.4 Value Side . . . 27
3.4.1 Grouping of Configuration Values . . . 30
3.4.2 Parameter . . . 30
3.4.3 Reference . . . 31
4 Details 32
4.1 Definition Side . . . 32
4.1.1 Parameter . . . 32
4.1.1.1 Enumeration Parameter . . . 32
4.1.1.2 Boolean Parameter . . . 34
4.1.1.3 Integer Parameter . . . 34
4.1.1.4 Float Parameter . . . 35
4.1.1.5 Unrestricted String Parameter . . . 37
4.1.1.6 Restricted String Parameter . . . 38
4.1.2 Reference . . . 43
4.1.2.1 Reference to Container . . . 44
4.1.2.2 Choice Reference to Container . . . 44

4.1.2.3 Foreign Reference . . . 45
4.1.2.4 Instance Reference . . . 47
4.2 Value Side . . . 51
4.2.1 Parameter . . . 51
4.2.1.1 Numerical Parameter Value . . . 52
4.2.1.2 Unrestricted Textual Parameter Value . . . 52
4.2.1.3 Restricted Textual Parameter Value . . . 53
4.2.2 Reference . . . 57
5 Modeling Use Cases 65
5.1 Functional Cluster interacts with Functional Cluster . . . 65
A Reference Material 67
A.1 Terms and Abbreviations . . . 67
A.2 Imposition Times of Constraints . . . 67
B Examples 69
C Change History of AUTOSAR Traceable Items 70
C.1 Traceable item history of this document according to AUTOSAR Release R24-11 . . . 70
C.1.1 Added Specification Items in R24-11 . . . 70
C.1.2 Changed Specification Items in R24-11 . . . 71
C.1.3 Deleted Specification Items in R24-11 . . . 71
C.1.4 Added Constraints in R24-11 . . . 72
C.1.5 Changed Constraints in R24-11 . . . 72
C.1.6 Deleted Constraints in R24-11 . . . 72
C.2 Traceable item history of this document according to AUTOSAR Release R25-11 . . . 72
C.2.1 Added Specification Items in R25-11 . . . 72
C.2.2 Changed Specification Items in R25-11 . . . 72
C.2.3 Deleted Specification Items in R25-11 . . . 73
C.2.4 Added Constraints in R25-11 . . . 73
C.2.5 Changed Constraints in R25-11 . . . 73
C.2.6 Deleted Constraints in R25-11 . . . 73
C.3 Traceable item history of this document according to AUTOSAR Release R26-11 . . . 74
C.3.1 Added Specification Items in R26-11 . . . 74
C.3.2 Changed Specification Items in R26-11 . . . 74
C.3.3 Deleted Specification Items in R26-11 . . . 74
C.3.4 Added Constraints in R26-11 . . . 74
C.3.5 Changed Constraints in R26-11 . . . 75
C.3.6 Deleted Constraints in R26-11 . . . 75
C.3.7 Added Advisories in R26-11 . . . 75
C.3.8 Changed Advisories in R26-11 . . . 76
C.3.9 Deleted Advisories in R26-11 . . . 76

D Mentioned Class Tables 77

E Non-Splitable Elements in the Scope of this Document 86

References

[1] Standardization Template
AUTOSAR_FO_TPS_StandardizationTemplate

[2] Specification of ECU Configuration
AUTOSAR_CP_TPS_ECUConfiguration

[3] Meta Model
AUTOSAR_FO_MMOD_MetaModel

[4] Meta Model-generated XML Schema
AUTOSAR_FO_MMOD_XMLSchema

[5] Generic Structure Template
AUTOSAR_FO_TPS_GenericStructureTemplate

[6] Glossary
AUTOSAR_FO_TR_Glossary

1 Introduction

This document describes the structure of the configuration (M2-level) model applicable to the creation of *target-configuration* (as opposed to *design-level*) content on the *AUTOSAR adaptive platform*.

The M2 model described in this document provides the structural basis for the creation of target-configuration (M1) models for individual functional clusters.

1.1 Document Conventions

Technical terms are typeset in mono spaced font, e.g. `PortPrototype`. As a general rule, plural forms of technical terms are created by adding “s” to the singular form, e.g. `PortPrototype`s. By this means the document resembles terminology used in the AUTOSAR XML Schema.

This document contains constraints in textual form that are distinguished from the rest of the text by a unique numerical constraint ID, a headline, and the actual constraint text starting after the ⌈ character and terminated by the ⌋ character.

The purpose of these constraints is to literally constrain the interpretation of the AUTOSAR meta-model such that it is possible to detect violations of the standardized behavior implemented in an instance of the meta-model (i.e. on M1 level).

Makers of AUTOSAR tools are encouraged to add the numerical ID of a constraint that corresponds to an M1 modeling issue as part of the diagnostic message issued by the tool.

The attributes of the classes introduced in this document are listed in form of class tables. They have the form shown in the example of the top-level element AUTOSAR:

Please note that constraints are not supposed to be enforceable at any given time in an AUTOSAR workflow. During the development of a model, constraints may legitimately be violated because an incomplete model will obviously show inconsistencies.

However, at specific points in the workflow, constraints shall be enforced as a safeguard against misconfiguration.

The points in the workflow where constraints shall be enforced, sometimes also known as the “imposition time” of the constraint, are different for each model category, e.g. on the classic platform, the constraints defined for software-components are typically enforced prior to the generation of the RTE while the constraints against the definition of an Ecu extract shall be applied when the Ecu configuration for the Com stack is created.

For each document, possible imposition times of constraints are defined and the imposition times are typically mentioned in the constraint themselves to give a proper orientation for implementers of AUTOSAR authoring tools.

Let AUTOSAR be an example of a typical class table. The first rows in the table have the following meaning:

Class: The name of the class as defined in the UML model.

Package: The UML package the class is defined in. This is only listed to help locating the class in the overall meta model.

Note: The comment the modeler gave for the class (class note). Stereotypes and UML tags of the class are also denoted here.

Base Classes: If applicable, the list of direct base classes.

The headers in the table have the following meaning:

Attribute: The name of an attribute of the class. Note that AUTOSAR does not distinguish between class attributes and owned association ends.

Type: The type of an attribute of the class.

Mul.: The assigned multiplicity of the attribute, i.e. how many instances of the given data type are associated with the attribute.

Kind: Specifies, whether the attribute is aggregated in the class (`aggr` aggregation), an UML attribute in the class (`attr` primitive attribute), or just referenced by it (`ref` reference). Instance references are also indicated (`iref` instance reference) in this field.

Note: The comment the modeler gave for the class attribute (role note). Stereotypes and UML tags of the class are also denoted here.

Please note that the chapters that start with a letter instead of a numerical value represent the appendix of the document. The purpose of the appendix is to support the explanation of certain aspects of the document and does not represent binding conventions of the standard.

The verbal forms for the expression of obligation specified in [TPS_STDT_00053] shall be used to indicate requirements, see [1, Standardization Template].

Please note: By intent, TPS documents (and their traceable items) do not trace up to an RS (requirement item).

2 Conceptual Aspects

2.1 Design and Target Configuration Models

If you look at AUTOSAR Models from a high vantage point, two kinds of models can be identified: design models and target-configuration models.

The purpose of this section is to give a rough intuition about the distinction between the two model flavors. It does not cover edge cases and messy details. While on first glance, this distinction is relatively obvious, there are also model elements that combine aspects of design and configuration. When looking at the details, the boundary is often blurry.

Please note that in the past, the term “deployment” was often used to refer to some aspects of “target-configuration”. “Deployment”, however, is heavily overloaded with many meanings.

To avoid confusion, it was decided to introduce the term “target-configuration”. It refers to configuration whose scope is local to a target, like ECU Configuration (EcuC) for an EcuInstance on the *AUTOSAR classic Platform*, and Adaptive Platform Machine Configuration (APMC) for a Machine on the *AUTOSAR adaptive Platform*.

2.1.1 Design Model

A rough criterion for identifying a model element as design-related, is whether or not the creation of the model element would typically be done in a System Designer role (usually related to the OEM scope).

Related to this point of view, is the question whether the model element contributes to making the overall system (beyond the scope of a single compute node) work.

In other words, a design-related model describes the external behavior, as in “the DEXT describes the external behavior of the Dcm/Dem/Fim (on the *AUTOSAR classic Platform*) or Dm (on the *AUTOSAR adaptive Platform*)”.

Examples for this perspective would be: the baud-rate of a network, or the definition of a CAN Id, or an application interface that acts as a contract between the application layer and the functional clusters.

As a consequence, many design-related model elements can’t be changed during the target-configuration phase, or else the system will break.

Another distinguishing characteristic of a design-related model element is that it is used to exchange information between different roles. For example between System Designer and Integrator, or between Application Developer and System Designer.

2.1.2 Target Configuration Models

The target-configuration model contains model elements whose scope is mostly local to a given compute node. Therefore, these elements are usually created during integration.

Some target-configuration-level model elements maintain a dedicated relation to design-level model elements, such that the target-configuration can only be considered complete, if the information from the design perspective is considered for the target-configuration.

There are also target-configuration elements that are directly derived from the content of a design-level model element. This strategy is heavily used on the *AUTOSAR classic Platform*.

It is not yet clarified whether it will also be relevant on the *AUTOSAR adaptive Platform*.

In contrast to the *AUTOSAR classic Platform* ECU Configuration model (EcuC), which often aims for a standalone model containing the complete information required to configure the BSW Stack, the APMC assumes that parts of the design model will be available on the target (usually not in ARXML format, but converted into a machine specific one).

For example, if a Service Interface can be used directly to configure an Adaptive Service, then there is no need to create a copy of that information in APMC.

2.2 Overview

The big picture of the target-configuration configuration on the *AUTOSAR adaptive Platform* looks very similar to the configuration on the *AUTOSAR classic Platform*, as described by the TPS ECU Configuration [2].

The difference is that the approach on the *AUTOSAR adaptive Platform* is simpler because it is not constrained by some of the boundary conditions that apply for the *AUTOSAR classic Platform*.

For example, APMC does not have to provide support for code generation (e.g. symbolic names), because this is currently not standardized on AP. In addition, the application of variation (let alone post-build variation) is not yet clarified on AP.

From high above, the configuration model differs from the design model such that the design model is build based on the AUTOSAR meta-model [3] (also known as the M2 model), whereas the configuration model follows the specification of the configuration parameters that in turn is also based on the AUTOSAR meta-model, but one meta-level below (also known as the M1 model).

As depicted by Figure 2.1, the difference between the M2 and the M1 model is that the M2 model is the source for the definition of (and serialization to) the XML schema [4]. On the other hand, an M1 model is serialized to an ARXML file.

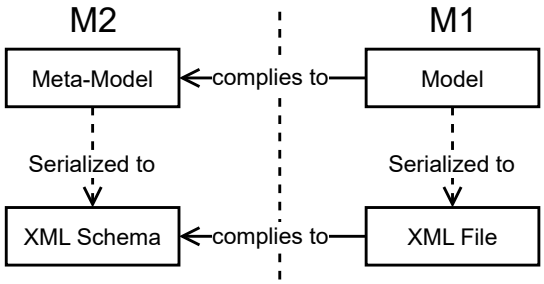


Figure 2.1: Visualization of different meta-levels from the serialization perspective

As also depicted in Figure 2.1, the M2 model has the job to describe the "grammar" in which the M1 model is formulated. An M1 model is expected to follow the structure defined by the corresponding M2 model.

On the serialization level, this means that the serialization of an M1 as an ARXML file shall validate against the serialization of the M2 model in the form of the AUTOSAR XML schema.

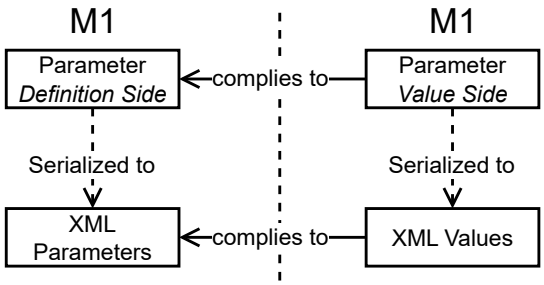


Figure 2.2: Visualization of the difference between parameter definition and parameter value

It is also possible to establish a relation between two models such that the first one establishes the "grammar" for the second model. This relation is depicted in Figure 2.2. On the file level, this means that the "value side" file shall follow the structure defined by the "definition side" file.

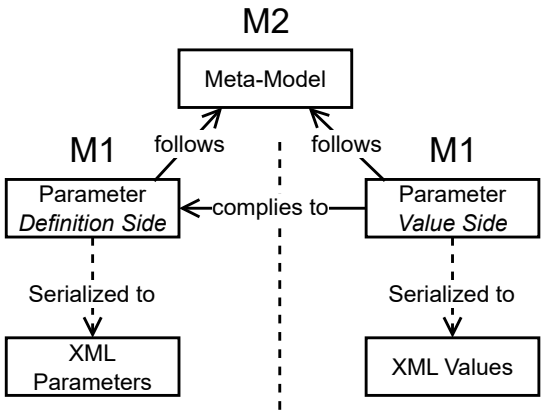


Figure 2.3: Visualization of the overall relation between meta-levels and "definition" vs. "value"

To summarize, the “value side” and the “definition side” on the M1 level have to follow the **structure** defined by the M2 model. In addition, the “value side” has to follow the **structure** of “definition side”. This aspect is depicted in Figure 2.3.

2.3 Benefits of using an M1 model-based configuration approach

In comparison to an equivalent implementation of the configuration model for the *AUTOSAR adaptive Platform* on the M2 level, the chosen approach comes with the following benefits:

- Much simpler model structure (easier to translate into vendor-specific `target-configuration` representations)
- Ability to define default values for specific configuration parameters
- Ability to create the model using the function-cluster-specific terminology
- Ability to define specific multiplicities (other than being restricted to 0..1 and 0..*)
- Ability to respond to changing requirements without having to wait for a schema change
- Ability to define vendor-specific extensions as first-class citizen
- Ability to respond to changing requirements without consideration of backwards compatibility on schema level
- Clear separation between the design-workflow and configuration-workflow

2.4 Modeling Strategies for deriving M1 Model Content

The configuration of the *AUTOSAR adaptive Platform* is not a stand-alone approach. Configuration models in many[1] cases need to refer to model elements from the design model in order to provide the complete picture.

In principle, different modeling strategies for combining the information from design model elements with the information on configuration level could be conceived.

2.4.1 Only Design exists

In some rare cases, it is possible to take the configuration of a functional cluster entirely from the design model, i.e. no configuration model is required in the first place.

[1] but most likely not all cases

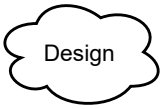


Figure 2.4: Visualization of the modeling strategy in which only a design model exists

2.4.2 Fully derive Design Model to Configuration Model

This modeling strategy is, to some extent, similar to the one described in Chapter 2.4.4, as its goal is to derive all relevant information from the design model into a respective representation in the configuration model.

After this derivation process completes, all relevant design information has become part of the configuration model. This means that the design model is no longer needed and technically might just as well be deleted.

This modeling strategy is very similar to the approach taken on the *AUTOSAR classic Platform* for the derivation of the configuration of the Com stack. While the derivation of the Com stack works very well, it has also been criticized for the amount of information duplication.

In other words, the configuration model tends to become big and complex if this modeling strategy is applied.

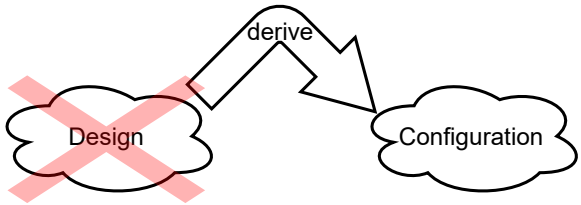


Figure 2.5: Visualization of the modeling strategy in which the design model is fully derived to the configuration

2.4.3 Partially derive Design Model to Configuration Model

If the full derivation of the design model to the configuration model, as described in Chapter 2.4.2 is considered to produce too complex models, a possible alternative could be to just derive the design model partially[2] and establish references to the still existing design model wherever these references can be modeled as flat references.

[2] In the extreme case: not at all, but this is a question of principle.

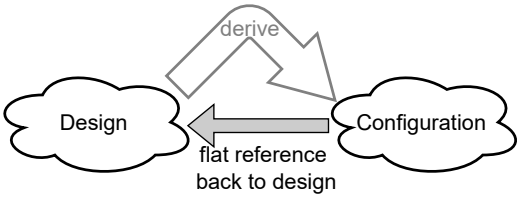


Figure 2.6: Visualization of the modeling strategy in which the configuration model is partially derived out of the design model and maintains flat references into the design model

Consequently, the part of the design model that would require the creation of flat references does not make use of the AUTOSAR type-prototype pattern.

The alternative to using flat references would be the usage of instanceRefs. Again, the derivation of the configuration model from the design model is optional.

The core characteristics of this modeling strategy is the utilization of instanceRefs, which also means that this modeling strategy is targeting the part of the meta-model where the type-prototype pattern has been used to create the meta-model.

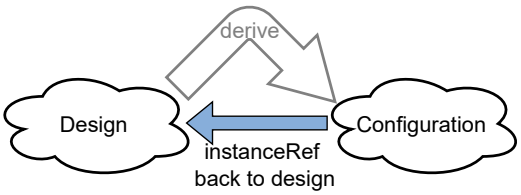


Figure 2.7: Visualization of the modeling strategy in which the configuration model is partially derived out of the design model and maintains instanceRefs into the design model

2.4.4 Only Configuration exists

This modeling strategy is based on the idea that there is no relevant design model, and all the configuration is exclusively done on the M1 level. Such a case is certainly special and does not represent a mainstream possibility.



Figure 2.8: Visualization of the modeling strategy in which only a configuration model exists

2.5 Vendor Specific Functional Cluster Definition

The Functional Cluster Definition that is standardized by AUTOSAR is the base for the configuration of the Functional Cluster. Out of the Standardized Functional Cluster Definition the Vendor-Specific Functional Cluster Definition can be derived.

Such a Vendor-Specific Functional Cluster Definition is allowed to define additional vendor-specific extensions (e.g. additional containers, parameters or references) with the same set of language elements as the Standardized Functional Cluster Definition.

This eases the development of tools and introduces the possibility to standardize vendor-specific parameters at a later point in time in AUTOSAR.

[AP_TPS_APMC_01038] `ApmcFunctionalClusterDef` categories ⌈The `category` attribute of `ApmcFunctionalClusterDef` defines the kind of the Functional Cluster Definition:

- `STANDARDIZED_CLUSTER_DEFINITION` - Functional Cluster Definition that is standardized and released by AUTOSAR
- `VENDOR_SPECIFIC_CLUSTER_DEFINITION` - Vendor Specific Cluster Definition that refines a `STANDARDIZED_CLUSTER_DEFINITION` with vendor specific extensions

⌋

[constr_9322] `ApmcFunctionalClusterDef` category restriction

Imposition time: `IT_CfgFcd`

⌈The following values for the attribute `category` of `ApmcFunctionalClusterDef` are reserved by the AUTOSAR Standard:

- `STANDARDIZED_CLUSTER_DEFINITION`
- `VENDOR_SPECIFIC_CLUSTER_DEFINITION`

⌋

Please note that custom values for attribute `category` attribute of `ApmcFunctionalClusterDef` are currently not supported.

2.6 Life Cycle definition

AUTOSAR provides support for life cycle handling, defined in the Generic Structure Template [5, AUTOSAR TPS Generic Structure Template]. A standardized usage of this approach is defined in the Standardization Template [1, AUTOSAR TPS Standardization Template].

For the definition of APMC Configuration Parameters there is support in the MetaModel to annotate the life cycle state of each ApmcDefinitionElement. For the annotation the following tagged value atp.Status can be used.

If a ApmcParamConfContainerDef defines the tag atp.Status, then the included aggregation of elements and containers are allowed to have the tag atp.Status set according to [AP_TPS_APMC_01040].

Within the table, a “1” represents an allowed combination and “0” represents a not-allowed combination.

[AP_TPS_APMC_01040] Matrix of allowed status value combinations of ApmcParamConfContainerDef and aggregations of ApmcParameterDef/ApmcReferenceDef/ApmcContainerDef ⌈

	Status of included aggregation			
Status of ApmcParamConfContainerDef	draft	valid	obsolete	removed
draft	1	0	1	1
valid	1	1	1	1
obsolete	0	0	1	1
removed	0	0	0	1

⌋

If an ApmcReferenceDef defines the tag atp.Status, then the target of this reference is allowed to use the tag atp.Status according to [AP_TPS_APMC_01041].

[AP_TPS_APMC_01041] Matrix of allowed status value combinations of referenced targets of a ApmcReferenceDef ⌈

	Status of reference target			
Status of ApmcReferenceDef	draft	valid	obsolete	removed
draft	1	1	0	0
valid	0	1	0	0
obsolete	1	1	1	0
removed	1	1	1	1

⌋

Please note that only values of atp.Status that are currently supported are “valid”, “obsolete” and “draft”. The life-cycle status “removed” is therefore equivalent to physical removal of the respective model element.

3 Overview

This chapter provides a description of the configuration elements with a focus on the overview and general modeling strategy.

As already hinted, the modeling approach for the `target-configuration` is split into two parts:

- The “definition side” is used to define the structure of the configuration for specific parts of the `target-configuration`, e.g. the configuration of a specific functional cluster.
- The “value side” is used to create values for specific configuration items according to the structure defined in the “definition side”.

In general, the structure of the configuration can be subdivided into two basic building blocks:

- containers,
- parameters and references.

Containers are used to group parameters, references and containers, thereby creating a nested configuration structure of arbitrary depth.

Parameters carry the relevant values that configure the specific parts of the `Machine` or `SoftwareCluster` assembly. References point to other containers that are relevant for the respective configuration context.

3.1 Abstract Structure

[AP_TPS_APMC_01000] Abstract relation between the “value side” and the “definition side” ⌈The relation between the “value side” and the “definition side” is that the “value side” needs to follow the structure defined by the “definition side”.

The concrete relation between a value and the corresponding definition is defined by a reference from the “value side” model element to the “definition side” model element.

This reference is established on an abstract level by the reference in the role `ApmcAbstractValue.definition` to `ApmcAbstractDefinition`. This reference is inherited by the respective corresponding sub-classes of `ApmcAbstractValue` and `ApmcAbstractDefinition`.⌋

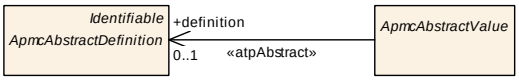


Figure 3.1: Abstract structure of the `target-configuration` configuration

Please note that there is no 1:1 relation between an `ApmcAbstractValue` and an `ApmcAbstractDefinition`. It is possible to create a collection with an upper multi-

plicity greater than 1 within the APMC model and therefore the same ApmcAbstract-Definition may be referenced by more than one ApmcAbstractValue.

Class	***ApmcAbstractDefinition*** (abstract)			
Note	This abstract meta-class implements the "definition" side of the abstract structure for the machine configuration. This Class is only used by the AUTOSAR Adaptive Platform.			
Base	*ARObject*, *Identifiable*, *MultilanguageReferrable*, *Referrable*			
Subclasses	*ApmcContainerDef*, *ApmcContainerElementDef*, ApmcFunctionalClusterDef			
Attribute	***Type***	***Mult.***	***Kind***	***Note***
–	–	–	–	–

Table 3.1: ApmcAbstractDefinition

Class	***ApmcAbstractValue*** (abstract)			
Note	This abstract meta-class implements the "value" side of the abstract structure for the machine configuration. This Class is only used by the AUTOSAR Adaptive Platform.			
Base	*ARObject*			
Subclasses	*ApmcContainerElementValue*, ApmcContainerValue, ApmcFunctionalClusterValue			
Attribute	***Type***	***Mult.***	***Kind***	***Note***
definition	ApmcAbstractDefinition	0..1	ref	This reference implements the relation of "value" side to "definition" side in the abstract structure of the machine configuration model. **Stereotypes:** atpAbstract

Table 3.2: ApmcAbstractValue

[constr_10579] Existence of definition

Imposition time: IT_CfgFcd

⌈In the context of an ApmcAbstractValue, the reference in the role definition shall exist.⌋

As a rough categorization, the following configuration elements are provided:

- Functional cluster
- Container
- Parameter
- Reference

An overview of the modeling on the “definition side” is given in Chapter 3.2. On the same level of detail, the modeling on the “value side” is described in Chapter 3.4.

3.2 Definition Side

[AP_TPS_APMC_01001] Root element on the "definition side" ⌈Meta-class ApmcFunctionalClusterDef represents the root element of the target-configuration configuration on the "definition side".⌋

[AP_TPS_APMC_01002] Semantics of ApmcDefinitionCollection ⌈Meta-class ApmcDefinitionCollection acts (as the name already suggests) as a collector, such that model elements that need to refer to a number of different ApmcFunctionalClusterDefs can instead simply create one reference to the ApmcDefinitionCollection that in turn refers to the affected ApmcFunctionalClusterDefs in the role functionalCluster.⌋

Class	**ApmcDefinitionCollection**			
Note	This class represents the entry point of the target-configuration on the "definition" side. **Tags:** atp.recommendedPackage=SoftwareClusterDefinitions This Class is only used by the AUTOSAR Adaptive Platform.			
Base	*ARElement*, *ARObject*, *AtpBlueprint*, *CollectableElement*, *Identifiable*, *MultilanguageReferrable*, *PackageableElement*, *Referrable*, *UploadablePackageElement*			
Aggregated by	ARPackage.element			
Attribute	***Type***	***Mult.***	***Kind***	***Note***
functional Cluster	ApmcFunctionalClusterDef	*	ref	This reference identifies the collection of functional clusters that are configured (partially) in the enclosing AmpcSoftwareClusterDefinition. **Tags:** atp.Splitkey=functionalCluster

Table 3.3: ApmcDefinitionCollection

Class	**ApmcFunctionalClusterDef**			
Note	This class represents the entry point into the configuration of a single functional cluster on the AUTOSAR adaptive platform on the "definition" side. **Tags:** atp.recommendedPackage=FunctionalClusterDefinitions This Class is only used by the AUTOSAR Adaptive Platform.			
Base	*ARElement*, *ARObject*, *ApmcAbstractDefinition*, *ApmcDefinitionElement*, *AtpBlueprint*, *Collectable Element*, *Identifiable*, *MultilanguageReferrable*, *PackageableElement*, *Referrable*, *UploadablePackage Element*			
Aggregated by	ARPackage.element			
Attribute	***Type***	***Mult.***	***Kind***	***Note***
container	ApmcContainerDef	*	aggr	This aggregation represents the collection of containers aggregated directly by the functional cluster configuration definition. **Tags:** atp.Splitkey=container, container.shortName

Table 3.4: ApmcFunctionalClusterDef

3.2.1 Definition Elements

Meta-class ApmcDefinitionElement is used to inherit some essential and commonly used attributes to the various classes on the “definition side” of the target-configuration.

Class	***ApmcDefinitionElement*** (abstract)			
Note	This abstract class contributes basic properties to the modeling of classes on the "definition" side of the target-configuration. This Class is only used by the AUTOSAR Adaptive Platform.			
Base	*ARObject*			
Subclasses	*ApmcContainerDef*, *ApmcContainerElementDef*, ApmcFunctionalClusterDef			
Attribute	***Type***	***Mult.***	***Kind***	***Note***
elementCreation	ApmcElementCreationEnum	0..1	attr	This attribute gives a hint about how (in terms of the AUTOSAR workflow) the model element comes into existence. **Tags:** xml.sequenceOffset=10
lowerMultiplicity	PositiveInteger	0..1	attr	This attribute specifies the lower multiplicity of the respective model element. **Tags:** xml.sequenceOffset=30
relatedTraceItem	Traceable	0..1	ref	This contains a sloppy reference to the formal ID of the element. **Stereotypes:** atpUriDef **Tags:** atp.Splitkey=relatedTraceItem xml.sequenceOffset=20
upperMultiplicity	PositiveInteger	0..1	attr	This attribute specifies the upper multiplicity of the respective model element. **Tags:** xml.sequenceOffset=40
upperMultiplicityIsInfinite	Boolean	0..1	attr	This attribute indicates (if set to true) that the upper multiplicity is set to "infinite". **Tags:** xml.sequenceOffset=50

Table 3.5: ApmcDefinitionElement

[constr_10643] Existence of attributes ApmcDefinitionElement.upperMultiplicity and ApmcDefinitionElement.upperMultiplicityIsInfinite

Imposition time: IT_CfgFcd

⌈For each ApmcDefinitionElement, **either** the attribute upperMultiplicity or upperMultiplicityIsInfinite shall exist.⌋

[AP_TPS_APMC_01003] Specification of element multiplicity on the “definition side” ⌈Contrary to the situation on the M2 model, multiplicities can’t be defined on an association, but on the associated element itself. For this purpose, the attributes

- ApmcDefinitionElement.lowerMultiplicity
- ApmcDefinitionElement.upperMultiplicity
- ApmcDefinitionElement.upperMultiplicityIsInfinite

are provided.

The purpose of attribute `upperMultiplicityIsInfinite` is to indicate that the upper multiplicity is not defined by the model element itself, but depends on the usage of the model element.

This intention cannot be communicated by the attribute `upperMultiplicity` because the latter expects a concrete value.⌋

Please note that the definition of multiplicities on the `target-configuration` is not subject to restrictions in terms of strict modeling patterns[1]. Any constellations are supported, as long as the value of `lowerMultiplicity` is less than or equal to the value `ApmcDefinitionElement.upperMultiplicity` if this attribute exists.

[constr_10580] Restriction regarding the value of `ApmcDefinitionElement.lowerMultiplicity` vs. `ApmcDefinitionElement.lowerMultiplicity`

Imposition time: `IT_CfgFcd`

⌈In the context of any given `ApmcDefinitionElement`, if attribute `upperMultiplicity` exists, then its value shall be greater than or equal to the value of the attribute `lowerMultiplicity` in the same context.⌋

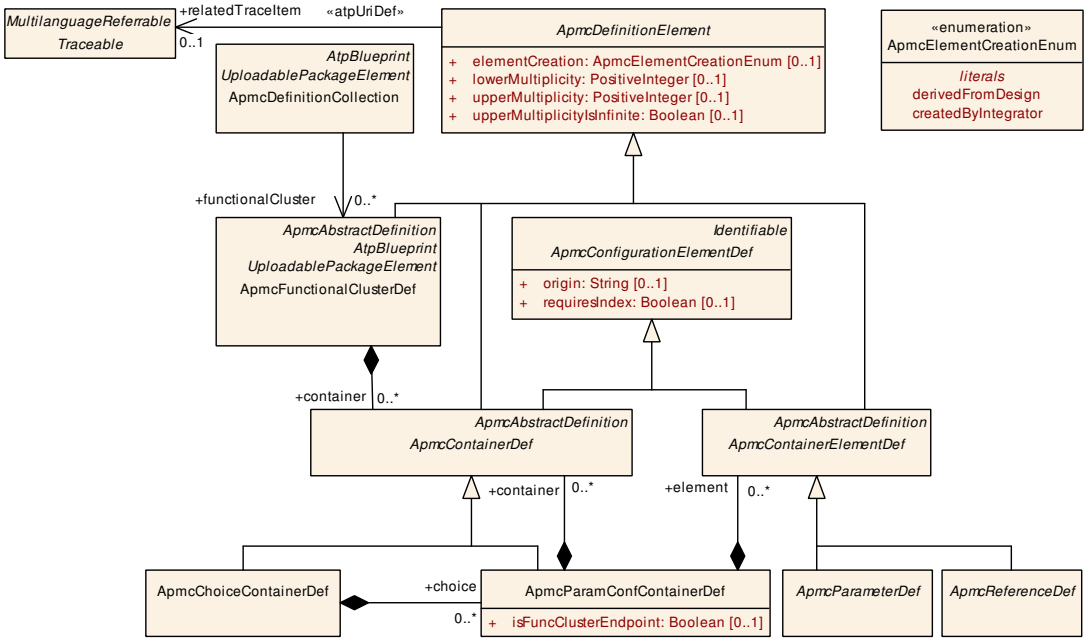


Figure 3.2: Overview of the `target-configuration` definition

[AP_TPS_APMC_01004] Semantics of attribute `ApmcDefinitionElement.elementCreation` ⌈Attribute `ApmcDefinitionElement.elementCreation` can be used to indicate the intended creation of the respective model element on the "definition side". The attribute allows for specifying the intention that either

[1] like on the M2 model

- the model element is foreseen to be derived from an equivalent model element in the design domain (in which case the value `ApmcElementCreationEnum.derivedFromDesign` applies) or
- the model element is intended to be created by an integrator (in which case the value `ApmcElementCreationEnum.createdByIntegrator` applies).

This information may be helpful for designers of tool workflows such that the handling of the parameters in the configuration workflow can be properly considered.⌋

Enumeration	**ApmcElementCreationEnum**
Note	This enumeration defines different approaches for the creation of a model element in the context of the target-configuration. This Enumeration is only used by the AUTOSAR Adaptive Platform.
Aggregated by	*ApmcDefinitionElement*.elementCreation
Literal	***Description***
createdByIntegrator	This literal indicates that the model element is explicitly created by an integrator during the deployment phase of the AUTOSAR workflow. **Tags:** atp.EnumerationLiteralIndex=1
derivedFromDesign	This literal indicates that the model element is derived from information available in the design step of the AUTOSAR workflow. **Tags:** atp.EnumerationLiteralIndex=0

Table 3.6: ApmcElementCreationEnum

[AP_TPS_APMC_01005] Semantics of attribute `ApmcConfigurationElementDef.origin` ⌈Attribute `ApmcConfigurationElementDef.origin` shall be used to indicate whether the respective model element has been defined as either

- a part of the AUTOSAR standard (in which case the value of the attribute shall be `AUTOSAR_APMC`) or
- as a vendor-specific configuration element (in which case the value of the attribute shall be clearly marked as vendor-specific by using the vendor name in the value of the attribute).

⌋

Class	***ApmcConfigurationElementDef*** (abstract)			
Note	This abstract class provides some attributes that are relevant as a contribution to the definition of meta-classes in the context the "definition" side of the target-configuration. This Class is only used by the AUTOSAR Adaptive Platform.			
Base	*ARObject*, *Identifiable*, *MultilanguageReferrable*, *Referrable*			
Subclasses	*ApmcContainerDef*, *ApmcContainerElementDef*			
Attribute	***Type***	***Mult.***	***Kind***	***Note***
origin	String	0..1	attr	This attribute provides the information whether the definition of the configuration element is standardized by AUTOSAR or whether it represents a vendor-specific configuration element.

▽

△

Class	***ApmcConfigurationElementDef*** (abstract)			
requiresIndex	Boolean	0..1	attr	This attribute indicates whether the elements of a collection of the enclosing model element carry an index value to allow for the identification of order. In other words, the order is not established on the syntactical level by the appearance in the ARXML files but on a semantic level by evaluating the value of the index attribute.

Table 3.7: ApmcConfigurationElementDef

In some cases, it is necessary to create collections where the order of elements in the collection is significant. On the M2 level, this can be achieved by using the qualifier "ordered" in the definition of the collection.

This ability does not exist on the M1 level and therefore the ordering is supported by the explicit definition of an index value for the elements of a collection.

[AP_TPS_APMC_01013] Semantics of attribute `ApmcConfigurationElementDef.requiresIndex` ⌈Attribute `ApmcConfigurationElementDef.requiresIndex` has been defined to allow for the creation of *ordered collections*. The position of an element of the collection within the collection is defined by an index value on the "value side".⌋

3.2.2 Grouping of Configuration Elements

The container level within the configuration approach exists for several purposes (see also Figure 3.2):

- Provide a simple grouping mechanism for other model elements, see `ApmcContainerElementDef`, resp. the aggregation in the role `ApmcParamConfContainerDef.element`.
- Provide a way to create nested structures of such a grouping, see `ApmcParamConfContainerDef.container`.
- Provide the ability to become a target for references that create a relation between containers on different nesting levels, see Chapter 4.1.
- Emulate "inheritance" by means of choice-containers, see `ApmcChoiceContainerDef`.

[AP_TPS_APMC_01006] Grouping of configuration items on the level of a container ⌈The grouping of configuration items is implemented by means of the roles:

- `ApmcParamConfContainerDef.element` for the aggregation of "primitive" configuration items.
- `ApmcParamConfContainerDef.container` for the creation of nested levels of containers.

⌋

Class	***ApmcContainerDef*** (abstract)			
Note	This abstract meta-class represents a container on the "definition" side of the target-configuration. This Class is only used by the AUTOSAR Adaptive Platform.			
Base	*ARObject*, *ApmcAbstractDefinition*, *ApmcConfigurationElementDef*, *ApmcDefinitionElement*, *Identifiable*, *MultilanguageReferrable*, *Referrable*			
Subclasses	ApmcChoiceContainerDef, ApmcParamConfContainerDef			
Aggregated by	ApmcFunctionalClusterDef.container, ApmcParamConfContainerDef.container			
Attribute	***Type***	***Mult.***	***Kind***	***Note***
–	–	–	–	–

Table 3.8: ApmcContainerDef

Class	***ApmcContainerElementDef*** (abstract)			
Note	This abstract meta-class represents an element of a container on the "definition" side of the target-configuration. This Class is only used by the AUTOSAR Adaptive Platform.			
Base	*ARObject*, *ApmcAbstractDefinition*, *ApmcConfigurationElementDef*, *ApmcDefinitionElement*, *Identifiable*, *MultilanguageReferrable*, *Referrable*			
Subclasses	*ApmcParameterDef*, *ApmcReferenceDef*			
Aggregated by	ApmcParamConfContainerDef.element			
Attribute	***Type***	***Mult.***	***Kind***	***Note***
–	–	–	–	–

Table 3.9: ApmcContainerElementDef

Class	**ApmcParamConfContainerDef**			
Note	This meta-class represents the definition of container (on the "definition" side of the target-configuration) that carries the definition of parameters and references. This Class is only used by the AUTOSAR Adaptive Platform.			
Base	*ARObject*, *ApmcAbstractDefinition*, *ApmcConfigurationElementDef*, *ApmcContainerDef*, *ApmcDefinitionElement*, *Identifiable*, *MultilanguageReferrable*, *Referrable*			
Aggregated by	ApmcChoiceContainerDef.choice, ApmcFunctionalClusterDef.container, ApmcParamConfContainerDef.container			
Attribute	***Type***	***Mult.***	***Kind***	***Note***
container	ApmcContainerDef	*	aggr	This aggregation represents the collection of sub-containers of the enclosing container. By using this aggregation arbitrary nested structures of containers can be created. **Tags:** atp.Splitkey=container, container.shortName
element	ApmcContainerElementDef	*	aggr	This aggregation represents the collection of elements in the container. **Tags:** atp.Splitkey=element, element.shortName
isFuncClusterEndpoint	Boolean	0..1	attr	This attribute indicates whether the enclosing ApmcParamConfContainerDef represents the configuration of an endpoint for the interaction with other functional clusters.

Table 3.10: ApmcParamConfContainerDef

3.2.3 Emulation of Inheritance

[AP_TPS_APMC_01007] Semantics of a choice container ⌈The existence of an ApmcChoiceContainerDef implements a structural alternative in the target-configuration. The behavior of the ApmcChoiceContainerDef has an effect similar to that of creating sub-classes on the M2 model.⌋

The statement made by [AP_TPS_APMC_01006] is visualized in Figure 3.3, in which the equivalent of an inheritance relation in the M2 model (left part of Figure 3.3) is created by means of the target-configuration M1 model (right part of Figure 3.3).

Consequently, the multiplicity of ApmcParamConfContainerDefs aggregated in the role ApmcChoiceContainerDef.choice is subject to a restriction, see [constr_10756].

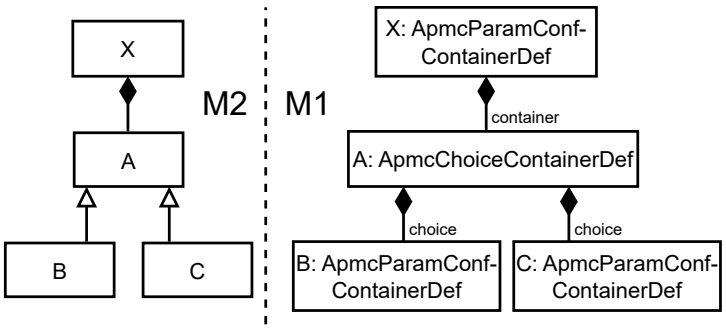


Figure 3.3: Visualization of the semantics of a choice container

[constr_10756] Multiplicity of ApmcParamConfContainerDefs aggregated in the role ApmcChoiceContainerDef.choice

Imposition time: IT_CfgFcd

⌈The multiplicity of each ApmcParamConfContainerDef aggregated in the role ApmcChoiceContainerDef.choice shall be set to 0..1.⌋

Class	**ApmcChoiceContainerDef**			
Note	This meta-class represents the ability to specify a choice-semantics on the container level (on the "definition" side of the target-configuration). The choice semantics can be utilized to emulate inheritance in the M1 model. This Class is only used by the AUTOSAR Adaptive Platform.			
Base	*ARObject*, *ApmcAbstractDefinition*, *ApmcConfigurationElementDef*, *ApmcContainerDef*, *ApmcDefinitionElement*, *Identifiable*, *MultilanguageReferrable*, *Referrable*			
Aggregated by	ApmcFunctionalClusterDef.container, ApmcParamConfContainerDef.container			
Attribute	***Type***	***Mult.***	***Kind***	***Note***
choice	ApmcParamConf ContainerDef	*	aggr	This collection of containers represents the content of the enclosing choice container. **Tags:** atp.Splitkey=choice, choice.shortName

Table 3.11: ApmcChoiceContainerDef

[advisory_01022] Existence of `ApmcChoiceContainerDef.choice`

Imposition time: `IT_CfgFcd`

⌈For each `ApmcChoiceContainerDef`, at least one `ApmcChoiceContainerDef.choice` should exist.⌋

3.2.4 Parameter

[AP_TPS_APMC_01008] Definition of configuration parameter ⌈A configuration parameter is used to adjust a specific atomic aspect of the `target-configuration`. Configuration parameters are available in different flavors, depending on the data type of the information they need to carry.

For this purpose, different data-type-specific subclasses of the abstract class `ApmcParameterDef` exist on the “definition side” to describe the definition of a parameter of a certain nature.

A configuration parameter can only be defined in the context of a `ApmcParamConfContainerDef`.⌋

The details are explained in Chapter 4.1.1.

The abstract meta-class `ApmcParameterDef` acts as a base class for the specific kinds of parameter definitions.

Class	***ApmcParameterDef*** (abstract)			
Note	This meta-class represents a parameter on the "definition" side of the adaptive platform. The concrete nature of the parameter is clarified by a subclass. This Class is only used by the AUTOSAR Adaptive Platform.			
Base	*ARObject*, *ApmcAbstractDefinition*, *ApmcConfigurationElementDef*, *ApmcContainerElementDef*, *ApmcDefinitionElement*, *Identifiable*, *MultilanguageReferrable*, *Referrable*			
Subclasses	*ApmcAbstractRestrictedStringParamDef*, ApmcBooleanParamDef, ApmcEnumerationParamDef, ApmcFloatParamDef, ApmcIntegerParamDef, ApmcStringParamDef			
Aggregated by	ApmcParamConfContainerDef.element			
Attribute	***Type***	***Mult.***	***Kind***	***Note***
–	–	–	–	–

Table 3.12: ApmcParameterDef

3.2.5 Reference

[AP_TPS_APMC_01009] Definition of reference ⌈In the context of the `target-configuration`, references can be used to either create a semantic link between different parts of the `target-configuration` or to associate a part of the `target-configuration` with an element from the design model.⌋

The details are explained in Chapter 4.1.2.

The abstract meta-class ApmcReferenceDef acts as a base class for the specific kinds of reference definition.

Class	***ApmcReferenceDef*** (abstract)			
Note	This meta-class represents a reference on the "definition" side of the adaptive platform. The concrete nature of the reference is clarified by a subclass. This Class is only used by the AUTOSAR Adaptive Platform.			
Base	*ARObject*, *ApmcAbstractDefinition*, *ApmcConfigurationElementDef*, *ApmcContainerElementDef*, *ApmcDefinitionElement*, *Identifiable*, *MultilanguageReferrable*, *Referrable*			
Subclasses	*ApmcAbstractForeignReferenceDef*, *ApmcAbstractInstanceReferenceDef*, ApmcChoiceContainerReferenceDef, ApmcContainerReferenceDef			
Aggregated by	ApmcParamConfContainerDef.element			
Attribute	***Type***	***Mult.***	***Kind***	***Note***
–	–	–	–	–

Table 3.13: ApmcReferenceDef

3.3 Reference to Traceable Elements

Meta-class ApmcDefinitionElement provides the ability to create a reference to a traceable element. This is achieved by means of the role ApmcDefinitionElement.relatedTraceItem.

The purpose of this reference is to allow for linking a specific ApmcDefinitionElement to the applicable SWS trace ID.

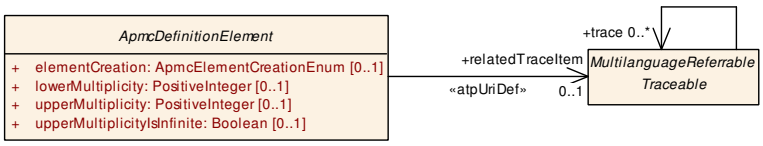


Figure 3.4: reference from ApmcDefinitionElement to a Traceable

3.4 Value Side

The value side of the target-configuration is where the actual configuration values are provided. Each configuration value is required to refer to the accompanying definition such that tools

- can interpret the provided value correctly with respect to the definition of the value and
- can check whether the nature of the provided value matches the expectation set by the corresponding element on the definition side.

The concrete references from “value side” to “definition side” with the semantics of referring to the defining configuration element are

- ApmcFunctionalClusterValue.definition

- ApmcContainerValue.definition
- ApmcContainerElementValue.definition

This modeling aspect is depicted in Figure 3.5.

[constr_10581] Existence of reference in the role ApmcFunctionalClusterValue.definition

Imposition time: IT_CfgFcd

⌈For each ApmcFunctionalClusterValue, the reference in the role definition shall exist.⌋

[constr_10582] Existence of reference in the role ApmcContainerValue.definition

Imposition time: IT_CfgFcd

⌈For each ApmcContainerValue, the reference in the role definition shall exist.⌋

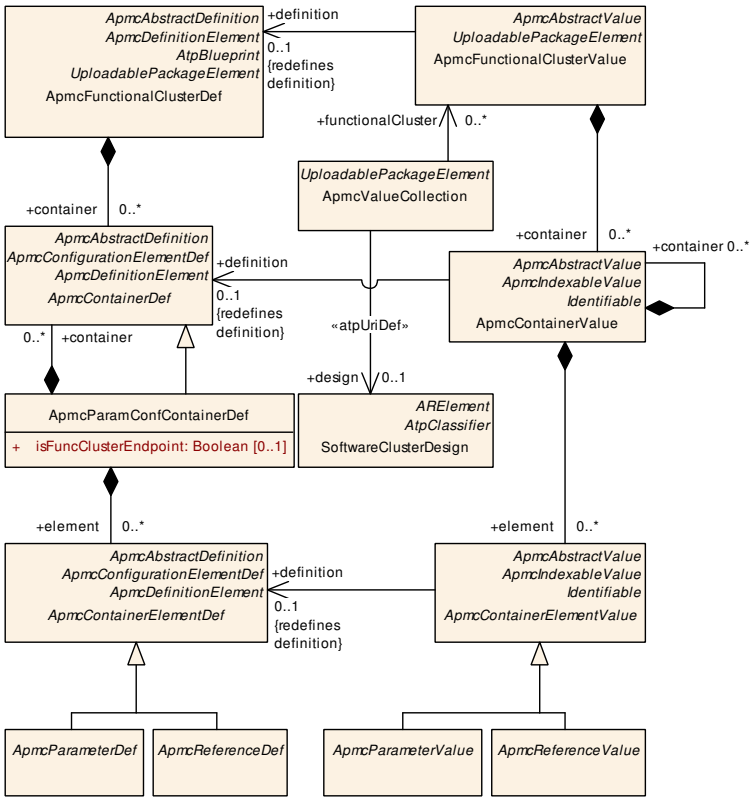


Figure 3.5: Overview of the target-configuration values

[constr_10583] Existence of reference in the role ApmcContainerElementValue.definition

Imposition time: IT_CfgFcd

⌈For each ApmcContainerElementValue, the reference in the role definition shall exist.⌋

[AP_TPS_APMC_01010] Root element on the "value side" ⌈The root element of the `target-configuration` on the "value side" is represented by meta-class `ApmcFunctionalClusterValue`.⌋

[AP_TPS_APMC_01036] Semantics of `ApmcValueCollection` ⌈Meta-class `ApmcValueCollection` acts (as the name already suggests) like a collector, such that model elements that need to refer to a number of different `ApmcFunctionalClusterValue`s can instead simply create one reference to the `ApmcValueCollection` that in turn refers to the affected `ApmcFunctionalClusterValue`s in the role `functionalCluster`.⌋

Class	**ApmcValueCollection**			
Note	This class represents the entry point of the target-configuration on the "value" side. **Tags:** atp.recommendedPackage=SoftwareClusterValues This Class is only used by the AUTOSAR Adaptive Platform.			
Base	*ARElement*, *ARObject*, *CollectableElement*, *Identifiable*, *MultilanguageReferrable*, *Packageable Element*, *Referrable*, *UploadablePackageElement*			
Aggregated by	ARPackage.element			
Attribute	***Type***	***Mult.***	***Kind***	***Note***
design	SoftwareClusterDesign	0..1	ref	This reference identifies the corresponding Software ClusterDesign. **Stereotypes:** atpUriDef **Tags:** atp.Splitkey=design
functional Cluster	ApmcFunctionalCluster Value	*	ref	This reference identifies the collection of functional clusters that are configured (partially) in the enclosing AmpcValueCollection. **Tags:** atp.Splitkey=functionalCluster

Table 3.14: ApmcValueCollection

Class	**ApmcFunctionalClusterValue**			
Note	This class represents the entry point into the configuration of a single functional cluster on the AUTOSAR adaptive platform on the "value" side. **Tags:** atp.recommendedPackage=FunctionalClusterValues This Class is only used by the AUTOSAR Adaptive Platform.			
Base	*ARElement*, *ARObject*, *ApmcAbstractValue*, *CollectableElement*, *Identifiable*, *MultilanguageReferrable*, *PackageableElement*, *Referrable*, *UploadablePackageElement*			
Aggregated by	ARPackage.element			
Attribute	***Type***	***Mult.***	***Kind***	***Note***
container	ApmcContainerValue	*	aggr	This aggregation represents the collection of containers aggregated directly by the functional cluster configuration definition. **Tags:** atp.Splitkey=container, container.shortName
definition	ApmcFunctionalCluster Def	0..1	ref	This reference implements the concrete relation of functional cluster configuration on the "value" side to functional cluster configuration on the "definition" side in the abstract structure of the machine configuration model. **Tags:** atp.Splitkey=definition

Table 3.15: ApmcFunctionalClusterValue

3.4.1 Grouping of Configuration Values

The definition of grouping and nested structures of configuration elements, as described on the "definition side" in Chapter 3.2.2 is reflected in the "value side" as well.

For this purpose, meta-class `ApmcContainerValue` has the ability to aggregate itself, see Figure 3.5.

Class	**ApmcContainerValue**			
Note	This meta-class represents a container on the "value" side of the target-configuration. This Class is only used by the AUTOSAR Adaptive Platform.			
Base	*ARObject*, *ApmcAbstractValue*, *ApmcIndexableValue*, *Identifiable*, *MultilanguageReferrable*, *Referrable*			
Aggregated by	ApmcContainerValue.container, ApmcFunctionalClusterValue.container			
Attribute	***Type***	***Mult.***	***Kind***	***Note***
container	ApmcContainerValue	*	aggr	This aggregation represents the collection of sub-containers of the enclosing container. By using this aggregation arbitrary nested structures of containers can be created. **Tags:** atp.Splitkey=container, container.shortName
definition	ApmcContainerDef	0..1	ref	This reference implements the concrete relation of container configuration on the "value" side to container configuration on the "definition" side in the abstract structure of the machine configuration model. **Tags:** atp.Splitkey=definition
element	ApmcContainerElement Value	*	aggr	This aggregation represents the collection of elements in the container. **Tags:** atp.Splitkey=element, element.shortName

Table 3.16: ApmcContainerValue

Class	***ApmcContainerElementValue*** (abstract)			
Note	This abstract meta-class represents an element of a container on the "value" side of the target-configuration. This Class is only used by the AUTOSAR Adaptive Platform.			
Base	*ARObject*, *ApmcAbstractValue*, *ApmcIndexableValue*, *Identifiable*, *MultilanguageReferrable*, *Referrable*			
Subclasses	*ApmcParameterValue*, *ApmcReferenceValue*			
Aggregated by	ApmcContainerValue.element			
Attribute	***Type***	***Mult.***	***Kind***	***Note***
definition	ApmcContainerElement Def	0..1	ref	This reference implements the concrete relation of container element configuration on the "value" side to container element configuration on the "definition" side in the abstract structure of the machine configuration model. **Tags:** atp.Splitkey=definition

Table 3.17: ApmcContainerElementValue

3.4.2 Parameter

As explained in Chapter 3.2.4, a configuration parameter can be defined in different flavors. These flavors are also represented on the value side, as subclasses of the abstract base class `ApmcParameterValue`.

Please find more information about the details in Chapter 4.2.1.

Class	***ApmcParameterValue*** (abstract)			
Note	This meta-class represents a parameter on the "value" side of the adaptive platform. The concrete nature of the parameter is clarified by a subclass. This Class is only used by the AUTOSAR Adaptive Platform.			
Base	*ARObject*, *ApmcAbstractValue*, *ApmcContainerElementValue*, *ApmcIndexableValue*, *Identifiable*, *MultilanguageReferrable*, *Referrable*			
Subclasses	*ApmcAbstractRestrictedTextualParamValue*, ApmcNumericalParamValue, ApmcTextualParamValue			
Aggregated by	ApmcContainerValue.element			
Attribute	**Type**	**Mult.**	**Kind**	**Note**
–	–	–	–	–

Table 3.18: ApmcParameterValue

3.4.3 Reference

References are also represented on the “value side” by means of the abstract base class `ApmcReferenceValue`. Please find more information about the details in Chapter 4.2.2.

Class	***ApmcReferenceValue*** (abstract)			
Note	This meta-class represents a reference on the "value" side of the adaptive platform. The concrete nature of the reference is clarified by a subclass. This Class is only used by the AUTOSAR Adaptive Platform.			
Base	*ARObject*, *ApmcAbstractValue*, *ApmcContainerElementValue*, *ApmcIndexableValue*, *Identifiable*, *MultilanguageReferrable*, *Referrable*			
Subclasses	*ApmcAbstractInstanceReferenceValue*, *ApmcAbstractReferenceValue*, ApmcContainerReferenceValue			
Aggregated by	ApmcContainerValue.element			
Attribute	**Type**	**Mult.**	**Kind**	**Note**
–	–	–	–	–

Table 3.19: ApmcReferenceValue

4 Details

This chapter provides a description of the configuration elements with a focus on the details.

4.1 Definition Side

This section contains a detailed description of configuration elements on the “definition side”.

4.1.1 Parameter

The detailed definition of the modeling of parameter definitions is depicted in Figure 4.1.

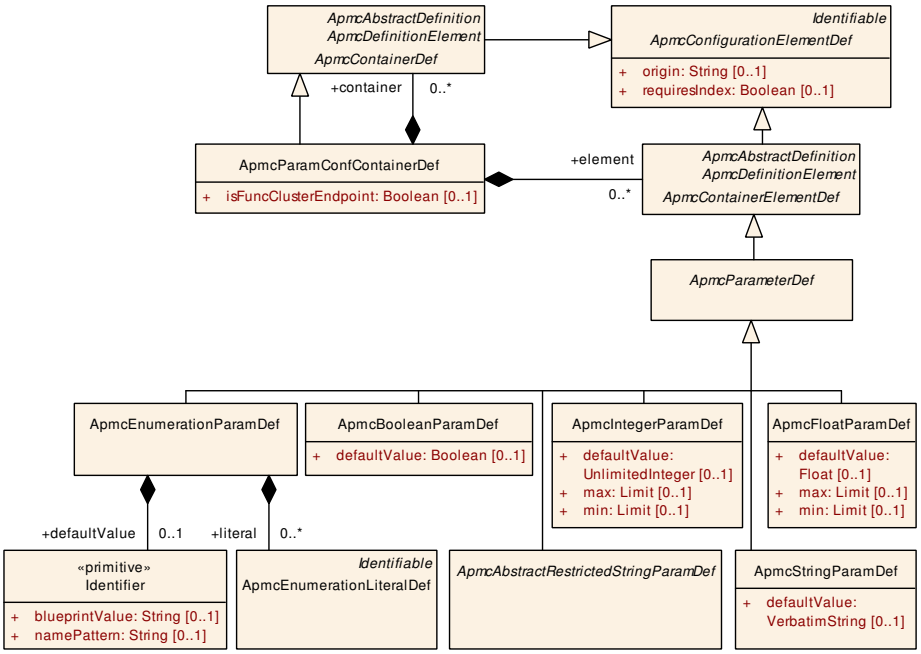


Figure 4.1: Overview of the manifest parameter definition

4.1.1.1 Enumeration Parameter

[AP_TPS_APMC_01011] Definition of an enumeration-style parameter ⌈The definition of an enumeration-style parameter is done by means of meta-classes `ApmcEnumerationParamDef` and `ApmcEnumerationLiteralDef`.

The former class is used to model the enumeration itself, while the latter class is used to model the specific values that the enumeration can take.

The literals that the enumeration can take are aggregated in the role `literal`.⌋

[AP_TPS_APMC_01042] Available choices of ApmcEnumerationParamDefs are defined by aggregated literals ⌈To provide the available choices for the ApmcEnumerationParamDef the ApmcEnumerationLiteralDef is used. Each available choice must have a corresponding literal defined.⌋

[AP_TPS_APMC_01043] The shortName of an ApmcEnumerationLiteralDef is used to define the literal ⌈The inherited shortName attribute is used to define the literals, so no additional attribute is needed.⌋

[AP_TPS_APMC_01012] Definition of default value for enumeration-style parameter ⌈It is possible to define a default value for an ApmcEnumerationParamDef, such that any parameter value that is created with a reference in the role definition can be created and immediately gets assigned a value taken from ApmcEnumerationParamDef.defaultValue.

The goal of the definition of a default value is to support creating multiple parameter values at a time, each of which is automatically assigned a value that reasonably approximates the final setting of the parameter in the sense that the user is "not wrong" if the parameter value is kept at the default value for the time being.

Of course, in the majority of cases, the default value will not remain and it will, at some point, be replaced by another value that is the result of the user's informed decision.⌋

Please note that [constr_10585] also applies for the definition of the ApmcEnumerationParamDef.defaultValue.

Note further that the class that corresponds to ApmcEnumerationParamDef on the "value side" is ApmcTextualParamValue.

Class	**ApmcEnumerationParamDef**			
Note	This meta-class represents an enumeration parameter on the "definition" side of the target-configuration. This Class is only used by the AUTOSAR Adaptive Platform.			
Base	*ARObject*, *ApmcAbstractDefinition*, *ApmcConfigurationElementDef*, *ApmcContainerElementDef*, *ApmcDefinitionElement*, *ApmcParameterDef*, *Identifiable*, *MultilanguageReferrable*, *Referrable*			
Aggregated by	ApmcParamConfContainerDef.element			
Attribute	***Type***	***Mult.***	***Kind***	***Note***
defaultValue	Identifier	0..1	attr	This attribute represents the definition of the default value for the enclosing parameter. The default value applies if the parameter is instantiated on the "value" side without assigning a dedicated value to it.
literal	ApmcEnumeration LiteralDef	*	aggr	This aggregation represents the collection of enumerators that are part of the enumeration. **Tags:** atp.Splitkey=literal, literal.shortName

Table 4.1: ApmcEnumerationParamDef

[constr_10757] Existence of the aggregation in the role `ApmcEnumerationParamDef.literal`

Imposition time: `IT_CfgFcd`

⌈For each `ApmcEnumerationParamDef`, at least one `ApmcEnumerationParamDef.literal` shall exist.⌋

Class	**ApmcEnumerationLiteralDef**			
Note	This meta-class represents an enumerator (or enumeration literal) on the "definition" side of the target-configuration. This Class is only used by the AUTOSAR Adaptive Platform.			
Base	*ARObject*, *Identifiable*, *MultilanguageReferrable*, *Referrable*			
Aggregated by	ApmcEnumerationParamDef.literal			
Attribute	***Type***	***Mult.***	***Kind***	***Note***
–	–	–	–	–

Table 4.2: ApmcEnumerationLiteralDef

4.1.1.2 Boolean Parameter

Meta-class `ApmcBooleanParamDef` is used to define a configuration parameter that carries a boolean value.

The class that corresponds to `ApmcBooleanParamDef` on the “value side” is `ApmcNumericalParamValue`.

For `ApmcBooleanParamDef`, it is possible to define a `defaultValue`, similar to [AP_TPS_APMC_01012]. The values used for the definition of the `defaultValue` shall follow the regulation imposed by [constr_10584].

Class	**ApmcBooleanParamDef**			
Note	This meta-class represents a boolean parameter on the "definition" side of the target-configuration. This Class is only used by the AUTOSAR Adaptive Platform.			
Base	*ARObject*, *ApmcAbstractDefinition*, *ApmcConfigurationElementDef*, *ApmcContainerElementDef*, *ApmcDefinitionElement*, *ApmcParameterDef*, *Identifiable*, *MultilanguageReferrable*, *Referrable*			
Aggregated by	ApmcParamConfContainerDef.element			
Attribute	***Type***	***Mult.***	***Kind***	***Note***
defaultValue	Boolean	0..1	attr	This attribute represents the definition of the default value for the enclosing parameter. The default value applies if the parameter is instantiated on the "value" side without assigning a dedicated value to it.

Table 4.3: ApmcBooleanParamDef

4.1.1.3 Integer Parameter

Meta-class `ApmcIntegerParamDef` is used to define a configuration parameter that carries an integer value.

The class that corresponds to ApmcIntegerParamDef on the “value side” is ApmcNumericalParamValue.

For ApmcIntegerParamDef, it is possible to define a defaultValue.

It is also possible to define the allowed range of the specific ApmcIntegerParamDef by means of attributes min and max.

[constr_10758] Restriction for Attributes ApmcIntegerParamDef.min and max

Imposition time: IT_CfgFcd

⌈For each ApmcIntegerParamDef, the value of attribute min shall be smaller than max.⌋

Class	**ApmcIntegerParamDef**			
Note	This meta-class represents an integer parameter on the "definition" side of the target-configuration. This Class is only used by the AUTOSAR Adaptive Platform.			
Base	*ARObject*, *ApmcAbstractDefinition*, *ApmcConfigurationElementDef*, *ApmcContainerElementDef*, *ApmcDefinitionElement*, *ApmcParameterDef*, *Identifiable*, *MultilanguageReferrable*, *Referrable*			
Aggregated by	ApmcParamConfContainerDef.element			
Attribute	***Type***	***Mult.***	***Kind***	***Note***
defaultValue	UnlimitedInteger	0..1	attr	This attribute represents the definition of the default value for the enclosing parameter. The default value applies if the parameter is instantiated on the "value" side without assigning a dedicated value to it.
max	Limit	0..1	attr	This attribute defines the upper limit of the parameter. This means that the value of the parameter shall never be greater than the defined upper limit.
min	Limit	0..1	attr	This attribute defines the lower limit of the parameter. This means that the value of the parameter shall never be less than the defined lower limit.

Table 4.4: ApmcIntegerParamDef

4.1.1.4 Float Parameter

Meta-class ApmcFloatParamDef is used to define a configuration parameter that carries a floating-point value.

The class that corresponds to ApmcFloatParamDef on the “value side” is ApmcNumericalParamValue.

For ApmcFloatParamDef, it is possible to define a defaultValue.

It is also possible to define the allowed range of the specific ApmcFloatParamDef by means of attributes min and max.

[constr_10759] Restriction for Attributes `ApmcFloatParamDef.min` and `max`

Imposition time: `IT_CfgFcd`

⌈For each `ApmcFloatParamDef`, the *numerical* (no consideration of interval type) value of attribute `min` shall be less than or equal to the *numerical* (no consideration of interval type) value of `max`.⌋

[AP_TPS_APMC_01044] Definition of interval type for `ApmcFloatParamDef.min` and `max` ⌈The attributes `ApmcFloatParamDef.min` and `max` are used to define the usable interval of the respective `ApmcFloatParamDef`. The interval itself may on both ends be defined as either

- closed: the provided value is included in the interval. This is expressed by setting the attribute `min.intervalType` resp. `max.intervalType` to `IntervalTypeEnum.closed`.
  - `min.intervalType` = `closed` is represented by **prepending** the interval with "["
  - `max.intervalType` = `closed` is represented by **appending** the interval with "]"
  - Example: Range: [0,1] = (`min.intervalType` = `closed` / `max.intervalType` = `closed`) means the range shall be (>=0 and <= 1).
- open: the provided value is not included in the interval. This is expressed by setting the attribute `min.intervalType` resp. `max.intervalType` to `IntervalTypeEnum.open`.
  - `min.intervalType` = `open` is represented by **prepending** the interval with "]"
  - `max.intervalType` = `open` is represented by **appending** the interval with "["
  - Example: Range:]0,1[= (`min.intervalType` = `open` / `max.intervalType` = `open`) means the range shall be (>0 and <1).

⌋

Class	**ApmcFloatParamDef**			
Note	This meta-class represents a float parameter on the "definition" side of the target-configuration. This Class is only used by the AUTOSAR Adaptive Platform.			
Base	*ARObject*, *ApmcAbstractDefinition*, *ApmcConfigurationElementDef*, *ApmcContainerElementDef*, *ApmcDefinitionElement*, *ApmcParameterDef*, *Identifiable*, *MultilanguageReferrable*, *Referrable*			
Aggregated by	ApmcParamConfContainerDef.element			
Attribute	***Type***	***Mult.***	***Kind***	***Note***

▽

△

Class	**ApmcFloatParamDef**			
defaultValue	Float	0..1	attr	This attribute represents the definition of the default value for the enclosing parameter. The default value applies if the parameter is instantiated on the "value" side without assigning a dedicated value to it.
max	Limit	0..1	attr	This attribute defines the upper limit of the parameter. This means that the value of the parameter shall never be greater than the defined upper limit.
min	Limit	0..1	attr	This attribute defines the lower limit of the parameter. This means that the value of the parameter shall never be less than the defined lower limit.

Table 4.5: ApmcFloatParamDef

Enumeration	**IntervalTypeEnum**
Note	This enumerator specifies the type of an interval.
Aggregated by	Limit.intervalType, LimitValueVariationPoint.intervalType
Literal	*Description*
closed	The area is limited by the value given. The value itself is included. **Tags:** atp.EnumerationLiteralIndex=0
open	The area is limited by the value given. The value itself is not included. **Tags:** atp.EnumerationLiteralIndex=2

Table 4.6: IntervalTypeEnum

4.1.1.5 Unrestricted String Parameter

Class	**ApmcStringParamDef**			
Note	This meta-class represents an string parameter on the "definition" side of the target-configuration. This Class is only used by the AUTOSAR Adaptive Platform.			
Base	*ARObject*, *ApmcAbstractDefinition*, *ApmcConfigurationElementDef*, *ApmcContainerElementDef*, *ApmcDefinitionElement*, *ApmcParameterDef*, *Identifiable*, *MultilanguageReferrable*, *Referrable*			
Aggregated by	ApmcParamConfContainerDef.element			
Attribute	*Type*	*Mult.*	*Kind*	*Note*
defaultValue	VerbatimString	0..1	attr	This attribute represents the definition of the default value for the enclosing parameter. The default value applies if the parameter is instantiated on the "value" side without assigning a dedicated value to it.

Table 4.7: ApmcStringParamDef

Meta-class `ApmcStringParamDef` is used to define a configuration parameter that carries a string value with unrestricted content.

The class that corresponds to `ApmcStringParamDef` on the “value side” is `ApmcTextualParamValue`.

For `ApmcStringParamDef`, it is possible to define a `defaultValue`.

4.1.1.6 Restricted String Parameter

There are some use cases for the definition of string-shaped configuration parameters where the content of the parameter has to follow a certain structure or pattern.

A prominent example of this kind of parameter could be the definition of an IP address. The string that carries the IP address is supposed to follow a convention for the definition of IP addresses.

It would therefore be good if it were possible to specify on the “definition side” that a certain parameter shall represent an IP address. This means that on the “value side” the content of the parameter shall follow the pre-defined convention.

In theory, it would be possible to define the restriction of a given parameter directly in the context of the definition of the parameter, i.e. by means of an attribute or tagged value owned by the meta-class `ApmcStringParamDef`.

However, such an approach clearly lacks the level of formal strictness that is provided by defining a specific meta-class for the specific use case, along with the definition of a regular expression that enforces the restriction on the string content.

This way, the restrictions uniformly apply to all instances of a specific use case and become immediately visible to consumers of a model.

Additionally, it is possible to validate the content of the various restriction use cases directly at the XML level by validating a serialized model against the AUTOSAR schema.

Here is the full list of use cases for restricted string content supported by this specification, along with the meta-class defined to support the use case:

- revision label, represented by `ApmcRevisionLabelParamDef`
- strong revision label, represented by `ApmcStrongRevisionLabelParamDef`
- IP V4 address, represented by `ApmcIpV4AddressParamDef`
- IP V6 address, represented by `ApmcIpV6AddressParamDef`
- MAC address, represented by `ApmcMacAddressParamDef`
- URI, represented by `ApmcUriParamDef`

The meta-classes for the definition of restricted string content are derived from the abstract class `ApmcAbstractRestrictedStringParamDef`.

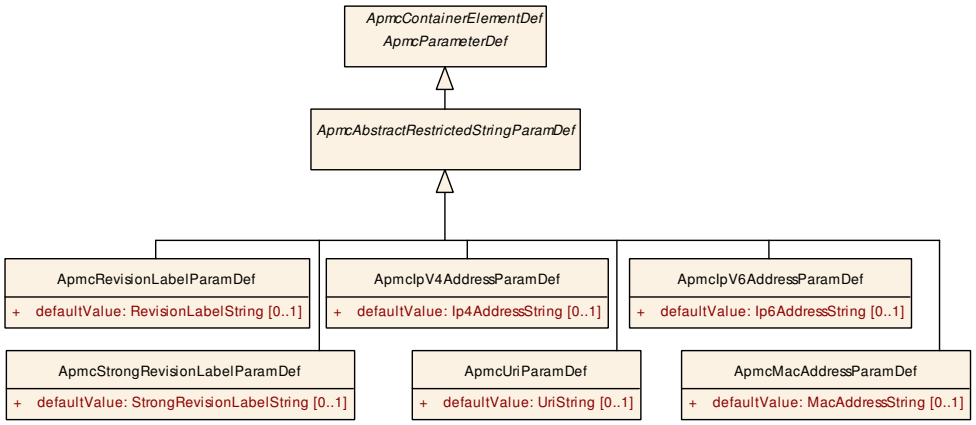


Figure 4.2: Modeling of the definition of string parameters with a content restriction

Each definition of a restricted string parameter shall support specifying a default value that conforms to the individual applicable restriction.

This needs to be achieved by the definition of a respective attribute that implements the restriction within each of the individual sub-classes of `ApmcAbstractRestricted-StringParamDef`, rather than defining the attribute in the abstract base class. This modeling is depicted in Figure 4.2.

Class	***ApmcAbstractRestrictedStringParamDef*** (abstract)			
Note	This meta-class represents an abstract base class for the definition of a string parameter where the content shall follow a structural restriction on the "definition" side of the target-configuration. The concrete restrictions are defined by sub-classes. This Class is only used by the AUTOSAR Adaptive Platform.			
Base	*ARObject*, *ApmcAbstractDefinition*, *ApmcConfigurationElementDef*, *ApmcContainerElementDef*, *ApmcDefinitionElement*, *ApmcParameterDef*, *Identifiable*, *MultilanguageReferrable*, *Referrable*			
Subclasses	ApmcIpV4AddressParamDef, ApmcIpV6AddressParamDef, ApmcMacAddressParamDef, ApmcRevisionLabelParamDef, ApmcStrongRevisionLabelParamDef, ApmcUriParamDef			
Aggregated by	ApmcParamConfContainerDef.element			
Attribute	***Type***	***Mult.***	***Kind***	***Note***
–	–	–	–	–

Table 4.8: ApmcAbstractRestrictedStringParamDef

Class	**ApmcRevisionLabelParamDef**
Note	This meta-class represents an restricted string parameter for the specification of a revision label on the "definition" side of the target-configuration. • supports three integers representing from left to right MajorVersion, MinorVersion, PatchVersion. • may add an application specific suffix separated by one of ".", "_", ";". Legal patterns are for example: • 4.0.0 • 4.0.0.1234565 • 4.0.0_vendor specific;13 • 4.0.0;12 This Class is only used by the AUTOSAR Adaptive Platform.

△

Class	**ApmcRevisionLabelParamDef**			
Base	*ARObject*, *ApmcAbstractDefinition*, *ApmcAbstractRestrictedStringParamDef*, *ApmcConfigurationElementDef*, *ApmcContainerElementDef*, *ApmcDefinitionElement*, *ApmcParameterDef*, *Identifiable*, *MultilanguageReferrable*, *Referrable*			
Aggregated by	ApmcParamConfContainerDef.element			
Attribute	***Type***	***Mult.***	***Kind***	***Note***
defaultValue	RevisionLabelString	0..1	attr	This attribute implements the default value of the revision label.

Table 4.9: ApmcRevisionLabelParamDef

Primitive	**RevisionLabelString**
Note	This primitive represents an internal AUTOSAR revision label which identifies an engineering object. It represents a pattern which • supports three integers representing from left to right MajorVersion, MinorVersion, PatchVersion. • may add an application specific suffix separated by one of ".", "_", ";". Legal patterns are for example: • 4.0.0 • 4.0.0.1234565 • 4.0.0_vendor specific;13 • 4.0.0;12 **Tags:** xml.xsd.customType=REVISION-LABEL-STRING xml.xsd.pattern=[0-9]+\.[0-9]+\.[0-9]+([\._;].*)? xml.xsd.type=string

Table 4.10: RevisionLabelString

Class	**ApmcStrongRevisionLabelParamDef**			
Note	This meta-class represents an restricted string parameter for the specification of a strong revision label on the "definition" side of the target-configuration. This class enforces a pattern which requires three integer numbers separated by a dot, representing from left to right MajorVersion, MinorVersion, PatchVersion and additional labels for pre-release version and build metadata. Legal patterns are for example: 1.0.0-alpha+001 1.0.0+20130313144700 1.0.0-beta+exp.sha.5114f85 This Class is only used by the AUTOSAR Adaptive Platform.			
Base	*ARObject*, *ApmcAbstractDefinition*, *ApmcAbstractRestrictedStringParamDef*, *ApmcConfigurationElementDef*, *ApmcContainerElementDef*, *ApmcDefinitionElement*, *ApmcParameterDef*, *Identifiable*, *MultilanguageReferrable*, *Referrable*			
Aggregated by	ApmcParamConfContainerDef.element			
Attribute	***Type***	***Mult.***	***Kind***	***Note***
defaultValue	StrongRevisionLabelString	0..1	attr	This attribute implements the default value of the strong revision label.

Table 4.11: ApmcStrongRevisionLabelParamDef

Primitive	**StrongRevisionLabelString**
Note	This primitive represents a revision label which identifies an object under version control. It represents a pattern which requires three integer numbers separated by a dot, representing from left to right Major Version, MinorVersion, PatchVersion and additional labels for pre-release version and build metadata. Legal patterns are for example: 1.0.0-alpha+001 1.0.0+20130313144700 1.0.0-beta+exp.sha.5114f85 **Tags:** xml.xsd.customType=STRONG-REVISION-LABEL-STRING xml.xsd.pattern=(0|[1-9]\d*)\.(0|[1-9]\d*)\.(0|[1-9]\d*)(-((0|[1-9]\d*|\d*[a-zA-Z-][0-9a-z A-Z-]*)(\.(0|[1-9]\d*|\d*[a-zA-Z-][0-9a-zA-Z-]*))*))?(\+([0-9a-zA-Z-]+(\.[0-9a-zA-Z-]+)*))? xml.xsd.type=string

Table 4.12: StrongRevisionLabelString

Class	**ApmcIpV4AddressParamDef**			
Note	This meta-class represents an restricted string parameter for the specification of an IP V4 address (notation: 255.255.255.255) on the "definition" side of the target-configuration. **Tags:** xml.name=APMC-IP-V4-ADDRESS-PARAM-DEF This Class is only used by the AUTOSAR Adaptive Platform.			
Base	*ARObject*, *ApmcAbstractDefinition*, *ApmcAbstractRestrictedStringParamDef*, *ApmcConfiguration ElementDef*, *ApmcContainerElementDef*, *ApmcDefinitionElement*, *ApmcParameterDef*, *Identifiable*, *MultilanguageReferrable*, *Referrable*			
Aggregated by	ApmcParamConfContainerDef.element			
Attribute	***Type***	***Mult.***	***Kind***	***Note***
defaultValue	Ip4AddressString	0..1	attr	This attribute implements the default value of the IP V4 address.

Table 4.13: ApmcIpV4AddressParamDef

Primitive	**Ip4AddressString**
Note	This is used to specify an IP4 address. Notation: 255.255.255.255 **Tags:** xml.xsd.customType=IP4-ADDRESS-STRING xml.xsd.pattern=(25[0-5]|2[0-4][0-9]|[01]?[0-9][0-9]?)\.(25[0-5]|2[0-4][0-9]|[01]?[0-9][0-9]?)\.(25[0-5]|2[0-4][0-9]|[01]?[0-9][0-9]?)\.(25[0-5]|2[0-4][0-9]|[01]?[0-9][0-9]?)|ANY xml.xsd.type=string

Table 4.14: Ip4AddressString

Class	**ApmcIpV6AddressParamDef**			
Note	This meta-class represents an restricted string parameter for the specification of an IP V4 address (notation: 255.255.255.255) on the "definition" side of the target-configuration. Notation: FFFF:FFFF:FFFF:FFFF:FFFF:FFFF:FFFF:FFFF Alternative notations, short-cuts with duplicate colons like ::, etc. or mixtures using colons and dots, are not allowed. **Tags:** xml.name=APMC-IP-V6-ADDRESS-PARAM-DEF This Class is only used by the AUTOSAR Adaptive Platform.			
Base	*ARObject*, *ApmcAbstractDefinition*, *ApmcAbstractRestrictedStringParamDef*, *ApmcConfiguration ElementDef*, *ApmcContainerElementDef*, *ApmcDefinitionElement*, *ApmcParameterDef*, *Identifiable*, *MultilanguageReferrable*, *Referrable*			
Aggregated by	ApmcParamConfContainerDef.element			
Attribute	***Type***	***Mult.***	***Kind***	***Note***
defaultValue	Ip6AddressString	0..1	attr	This attribute implements the default value of the IP V6 address.

Table 4.15: ApmcIpV6AddressParamDef

Primitive	**Ip6AddressString**
Note	This is used to specify an IP6 address. Notation: FFFF:FFFF:FFFF:FFFF:FFFF:FFFF:FFFF:FFFF Alternative notations, short-cuts with duplicate colons like ::, etc. or mixtures using colons and dots, are not allowed. **Tags:** xml.xsd.customType=IP6-ADDRESS-STRING xml.xsd.pattern=[0-9A-Fa-f]{1,4}(:[0-9A-Fa-f]{1,4}){7,7}|ANY xml.xsd.type=string

Table 4.16: Ip6AddressString

Class	**ApmcMacAddressParamDef**			
Note	This meta-class represents an restricted string parameter for the specification of a MAC address label on the "definition" side of the target-configuration. Notation: FF:FF:FF:FF:FF:FF Alternative notations, e.g. using dash instead of colon, or another grouping of numbers, is not allowed. This Class is only used by the AUTOSAR Adaptive Platform.			
Base	*ARObject*, *ApmcAbstractDefinition*, *ApmcAbstractRestrictedStringParamDef*, *ApmcConfiguration ElementDef*, *ApmcContainerElementDef*, *ApmcDefinitionElement*, *ApmcParameterDef*, *Identifiable*, *MultilanguageReferrable*, *Referrable*			
Aggregated by	ApmcParamConfContainerDef.element			
Attribute	***Type***	***Mult.***	***Kind***	***Note***
defaultValue	MacAddressString	0..1	attr	This attribute implements the default value of the MAC address.

Table 4.17: ApmcMacAddressParamDef

Primitive	**MacAddressString**
Note	This primitive specifies a Mac Address. Notation: FF:FF:FF:FF:FF:FF Alternative notations, e.g. using dash instead of colon, or another grouping of numbers, is not allowed. **Tags:** xml.xsd.customType=MAC-ADDRESS-STRING xml.xsd.pattern=([0-9a-fA-F]{2}:){5}[0-9a-fA-F]{2} xml.xsd.type=string

Table 4.18: MacAddressString

Class	**ApmcUriParamDef**			
Note	This meta-class represents an restricted string parameter for the specification of a Uniform Resource Identifier (URI) label on the "definition" side of the target-configuration. A Uniform Resource Identifier (URI), is a compact string of characters used to identify or name a resource. This Class is only used by the AUTOSAR Adaptive Platform.			
Base	*ARObject*, *ApmcAbstractDefinition*, *ApmcAbstractRestrictedStringParamDef*, *ApmcConfiguration ElementDef*, *ApmcContainerElementDef*, *ApmcDefinitionElement*, *ApmcParameterDef*, *Identifiable*, *MultilanguageReferrable*, *Referrable*			
Aggregated by	ApmcParamConfContainerDef.element			
Attribute	***Type***	***Mult.***	***Kind***	***Note***
defaultValue	UriString	0..1	attr	This attribute implements the default value of the URI.

Table 4.19: ApmcUriParamDef

Primitive	**UriString**
Note	A Uniform Resource Identifier (URI), is a compact string of characters used to identify or name a resource. **Tags:** xml.xsd.customType=URI-STRING xml.xsd.type=string

Table 4.20: UriString

4.1.2 Reference

The detailed modeling of references on the definition side is depicted in Figure 4.3.

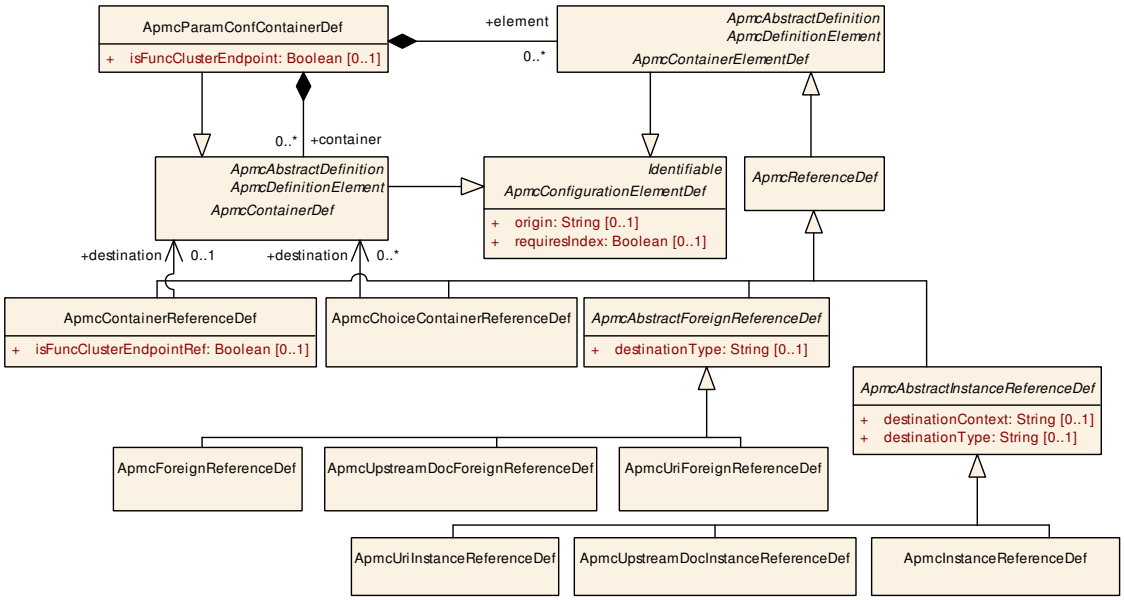


Figure 4.3: Overview of the manifest reference definition

The modeling supports four kinds of references:

- Reference to a container (see Chapter 4.1.2.1)
- A choice reference that resolves to one of the possible destination targets (see Chapter 4.1.2.2)
- Reference to an element of the M2 model (so-called *foreign* reference, see Chapter 4.1.2.3)
- Instance Reference to an element of the M2 model (so-called *instanceRef*, see Chapter 4.1.2.4)

4.1.2.1 Reference to Container

Within the M1 model, it is not possible to reference elements represented by abstract meta-class ApmcContainerElementDef. The only valid target of a reference is an instance of abstract base class ApmcContainerDef.

[AP_TPS_APMC_01014] Semantics of meta-class ApmcContainerReferenceDef ⌈Meta-class ApmcContainerReferenceDef shall be used if a reference to one ApmcContainerDef needs to be modeled.

The usage of meta-class ApmcContainerReferenceDef is independent of whether the target of the reference is defined in the scope of the same ApmcFunctionalClusterDef or not.⌋

[AP_TPS_APMC_01015] Semantics of reference ApmcContainerReferenceDef.destination ⌈Attribute ApmcContainerReferenceDef.destination is supposed to be filled with a fully-qualified shortName of the reference target.⌋

Class	**ApmcContainerReferenceDef**			
Note	This meta-class represents a reference to another container of the target-configuration. This Class is only used by the AUTOSAR Adaptive Platform.			
Base	*ARObject*, *ApmcAbstractDefinition*, *ApmcConfigurationElementDef*, *ApmcContainerElementDef*, *ApmcDefinitionElement*, *ApmcReferenceDef*, *Identifiable*, *MultilanguageReferrable*, *Referrable*			
Aggregated by	ApmcParamConfContainerDef.element			
Attribute	***Type***	***Mult.***	***Kind***	***Note***
destination	ApmcContainerDef	0..1	ref	This reference identifies the target container. **Tags:** atp.Splitkey=destination
isFuncClusterEndpointRef	Boolean	0..1	attr	This attribute indicates whether the enclosing ApmcContainerReferenceDef identifies endpoint for the interaction with other functional clusters.

Table 4.21: ApmcContainerReferenceDef

4.1.2.2 Choice Reference to Container

[AP_TPS_APMC_01016] Semantics of meta-class ApmcChoiceContainerReferenceDef ⌈Meta-class ApmcContainerReferenceDef shall be used if a reference to more than one ApmcContainerDef needs to be modeled.

The semantics of the collection of references is ultimately, only one of the possible references shall exist on the “value side”.

The usage of meta-class ApmcChoiceContainerReferenceDef is independent of whether the target of the reference is defined in the scope of the same ApmcFunctionalClusterDef or not.⌋

Class	ApmcChoiceContainerReferenceDef			
Note	This meta-class represents the ability to establish a reference that points to a collection of targets, out of which only one element can be picked at any given time. This Class is only used by the AUTOSAR Adaptive Platform.			
Base	*ARObject*, *ApmcAbstractDefinition*, *ApmcConfigurationElementDef*, *ApmcContainerElementDef*, *ApmcDefinitionElement*, *ApmcReferenceDef*, *Identifiable*, *MultilanguageReferrable*, *Referrable*			
Aggregated by	ApmcParamConfContainerDef.element			
Attribute	**Type**	**Mult.**	**Kind**	**Note**
destination	ApmcContainerDef	*	ref	This represents the collection of potential reference targets. **Tags:** atp.Splitkey=destination

Table 4.22: ApmcChoiceContainerReferenceDef

4.1.2.3 Foreign Reference

The modeling of a “foreign”[1] reference is based on the definition of the abstract base-class `ApmcAbstractForeignReferenceDef`.

[AP_TPS_APMC_01017] Semantics of primitive attribute `ApmcAbstractForeignReferenceDef.destinationType` ⌈Attribute `ApmcAbstractForeignReferenceDef.destinationType` is supposed to be filled with an Identification of the class of the reference target.

The class name is supposed to be provided in “schema notation” (Example: use `PROCESS-DESIGN` as a representation of meta-class `ProcessDesign`).⌋

Class	*ApmcAbstractForeignReferenceDef* (abstract)			
Note	This abstract meta-class represents the ability to establish a flat (as opposed to an instanceRef) reference to model elements out of the context of target-configuration, i.e. in the design model. This Class is only used by the AUTOSAR Adaptive Platform.			
Base	*ARObject*, *ApmcAbstractDefinition*, *ApmcConfigurationElementDef*, *ApmcContainerElementDef*, *ApmcDefinitionElement*, *ApmcReferenceDef*, *Identifiable*, *MultilanguageReferrable*, *Referrable*			
Subclasses	ApmcForeignReferenceDef, ApmcUpstreamDocForeignReferenceDef, ApmcUriForeignReferenceDef			
Aggregated by	ApmcParamConfContainerDef.element			
Attribute	**Type**	**Mult.**	**Kind**	**Note**
destinationType	String	0..1	attr	This attribute carries the identification of the target’s class. This way, the possible reference targets can be restricted on the "definition" side of the target-configuration. This is necessary to preserve the intended semantics of the foreign reference.

Table 4.23: ApmcAbstractForeignReferenceDef

[1] in the sense that the target of the reference is not a part of the M1 model

[constr_10760] Existence of attribute ApmcAbstractForeignReferenceDef.destinationType

Imposition time: IT_CfgFcd

⌈For each ApmcAbstractForeignReferenceDef, the attribute destination-Type shall exist.⌋

[AP_TPS_APMC_01018] Semantics of ApmcForeignReferenceDef ⌈Meta-class ApmcForeignReferenceDef is used to establish a **flat** reference to a referable[2] model element on the M2 level.

In the case of the target-configuration, the M2 model elements are used to define design-related model content and therefore the ApmcForeignReferenceDef is factually used to **establish a reference from the configuration domain to the design domain**.

The target of the reference is expected to exist in the same model scope as the owner of the ApmcForeignReferenceDef.⌋

Class	**ApmcForeignReferenceDef**			
Note	This meta-class represents the ability to establish a flat (as opposed to an instanceRef) reference to model elements out of the context of target-configuration, i.e. reference the element in the design model that contributes to the semantics of the target-configuration model. This Class is only used by the AUTOSAR Adaptive Platform.			
Base	*ARObject*, *ApmcAbstractDefinition*, *ApmcAbstractForeignReferenceDef*, *ApmcConfigurationElement Def*, *ApmcContainerElementDef*, *ApmcDefinitionElement*, *ApmcReferenceDef*, *Identifiable*, *MultilanguageReferrable*, *Referrable*			
Aggregated by	ApmcParamConfContainerDef.element			
Attribute	***Type***	***Mult.***	***Kind***	***Note***
–	–	–	–	–

Table 4.24: ApmcForeignReferenceDef

[AP_TPS_APMC_01019] Semantics of meta-class ApmcUriForeignReference-Def ⌈In principle, a ApmcUriForeignReferenceDef fulfills the same role as a ApmcForeignReferenceDef.

The difference is that in the case of the ApmcUriForeignReferenceDef, the target of the **flat** reference is **not** required to exist in the same model scope.

On the "value side", if the target of the reference does not exist in the same model scope, then the existence of the ApmcUriForeignReferenceDef can serve as an **additional identifying element** of the enclosing ApmcContainerValue of the Apmc-UriForeignReferenceDef.⌋

[2] i.e. derived from meta-class Referrable

Class	ApmcUriForeignReferenceDef			
Note	This meta-class represents the ability to establish a flat (as opposed to an instanceRef) reference to model elements out of the context of target-configuration. The target of the reference is not required to exist. The reference itself is already considered an identifying element. This Class is only used by the AUTOSAR Adaptive Platform.			
Base	*ARObject*, *ApmcAbstractDefinition*, *ApmcAbstractForeignReferenceDef*, *ApmcConfigurationElement Def*, *ApmcContainerElementDef*, *ApmcDefinitionElement*, *ApmcReferenceDef*, *Identifiable*, *MultilanguageReferrable*, *Referrable*			
Aggregated by	ApmcParamConfContainerDef.element			
Attribute	Type	Mult.	Kind	Note
–	–	–	–	–

Table 4.25: ApmcUriForeignReferenceDef

[AP_TPS_APMC_01020] Semantics of meta-class ApmcUpstreamDocForeignReferenceDef ⌈The purpose of the ApmcUpstreamDocForeignReferenceDef is to support the creation of a **flat** reference to a model **element in the design domain from which the enclosing container was created by derivation**.

The existence of ApmcUpstreamDocForeignReferenceValue that refers to a ApmcUpstreamDocForeignReferenceDef in the role definition is considered pure metadata.

In contrast to ApmcForeignReferenceDef and ApmcUriForeignReferenceDef, the ApmcUpstreamDocForeignReferenceValue does not carry relevant model semantics and serves only to indicate how the enclosing container came into existence.⌋

Class	ApmcUpstreamDocForeignReferenceDef			
Note	This meta-class represents the ability to establish a flat (as opposed to an instanceRef) reference to model elements out of the context of target-configuration, i.e. reference the element in the design model that sparked the existence of the owner of the reference. This Class is only used by the AUTOSAR Adaptive Platform.			
Base	*ARObject*, *ApmcAbstractDefinition*, *ApmcAbstractForeignReferenceDef*, *ApmcConfigurationElement Def*, *ApmcContainerElementDef*, *ApmcDefinitionElement*, *ApmcReferenceDef*, *Identifiable*, *MultilanguageReferrable*, *Referrable*			
Aggregated by	ApmcParamConfContainerDef.element			
Attribute	Type	Mult.	Kind	Note
–	–	–	–	–

Table 4.26: ApmcUpstreamDocForeignReferenceDef

4.1.2.4 Instance Reference

The instance references described in this chapter support the structured references to a model tree in which the AUTOSAR type-prototype is implemented. These instance references will be realized by meta-class AnyInstanceRef in an AUTOSAR model.

[AP_TPS_APMC_01021] Semantics of attribute ApmcAbstractInstanceReferenceDef.destinationContext ⌈The attribute destinationContext of

meta-class ApmcAbstractInstanceReferenceDef establishes the collection of context references used to define the ApmcAbstractInstanceReferenceDef.⌋

[AP_TPS_APMC_01045] Content of attribute ApmcAbstractInstanceReferenceDef.destinationContext ⌈Attribute ApmcAbstractInstanceReferenceDef.destinationContext is supposed to be filled with an ordered list of M2 class names that represent the ordered list of contents for the respective ApmcAbstractInstanceReferenceDef.

The class names are supposed to be provided in "schema notation" (Example: use SW-COMPONENT-PROTOTYPE as a representation of meta-class SwComponentPrototype).

If a specific meta-class may appear multiple times in a row, then the '*' character shall be directly appended to the class name. In the case of SwComponentPrototype, the indication of an ordered collection of SwComponentPrototypes in a row would read SW-COMPONENT-PROTOTYPE*.⌋

[AP_TPS_APMC_01022] Semantics of attribute ApmcAbstractInstanceReferenceDef.destinationType ⌈The attribute destinationType of meta-class ApmcAbstractInstanceReferenceDef establishes the target reference used to define the ApmcAbstractInstanceReferenceDef.⌋

[AP_TPS_APMC_01046] Content of attribute ApmcAbstractInstanceReferenceDef.destinationType ⌈Attribute ApmcAbstractInstanceReferenceDef.destinationType is supposed to be filled with an Identification of the class of the instance reference target.

The class name is supposed to be provided in "schema notation" (Example: use PROCESS-DESIGN as a representation of meta-class ProcessDesign).⌋

Class	***ApmcAbstractInstanceReferenceDef*** (abstract)			
Note	This abstract meta-class represents the ability to establish a instanceRef (as opposed to flat) reference to model elements. Subclasses clarify the specific semantics. This Class is only used by the AUTOSAR Adaptive Platform.			
Base	*ARObject*, *ApmcAbstractDefinition*, *ApmcConfigurationElementDef*, *ApmcContainerElementDef*, *ApmcDefinitionElement*, *ApmcReferenceDef*, *Identifiable*, *MultilanguageReferrable*, *Referrable*			
Subclasses	ApmcInstanceReferenceDef, ApmcUpstreamDocInstanceReferenceDef, ApmcUriInstanceReferenceDef			
Aggregated by	ApmcParamConfContainerDef.element			
Attribute	***Type***	***Mult.***	***Kind***	***Note***
destination Context	String	0..1	attr	This attribute carries the ordered list (separated by whitespace) of identification of context classes. This way, the possible reference targets can be restricted on the "definition" side of the target-configuration. This is necessary to preserve the intended semantics of the instance reference.

▽

△

Class	***ApmcAbstractInstanceReferenceDef*** (abstract)			
destinationType	String	0..1	attr	This attribute carries the identification of the target's class. This way, the possible reference targets can be restricted on the "definition" side of the target-configuration. This is necessary to preserve the intended semantics of the instance reference.

Table 4.27: ApmcAbstractInstanceReferenceDef

[constr_10761] Existence of attribute `ApmcAbstractInstanceReferenceDef.destinationContext`

Imposition time: `IT_CfgFcd`

⌈For each `ApmcAbstractInstanceReferenceDef`, the attribute `destinationContext` shall exist.⌋

[constr_10762] Existence of attribute `ApmcAbstractInstanceReferenceDef.destinationType`

Imposition time: `IT_CfgFcd`

⌈For each `ApmcAbstractInstanceReferenceDef`, the attribute `destinationType` shall exist.⌋

[AP_TPS_APMC_01023] Semantics of `ApmcInstanceReferenceDef` ⌈Meta-class `ApmcInstanceReferenceDef` is used to establish a **structured** reference to a referable[3] model element on the M2 level.

In the case of the `target-configuration`, the M2 model elements are used to define design-related model content and therefore the `ApmcInstanceReferenceDef` is factually used to **establish a reference from the configuration domain to the design domain**.

All context references and the target reference are expected to exist in the same model scope as the owner of the `ApmcInstanceReferenceDef`.⌋

Class	**ApmcInstanceReferenceDef**			
Note	This meta-class represents the ability to establish a instanceRef (as opposed to flat) reference to model elements out of the context of target-configuration, i.e. in the design model. This Class is only used by the AUTOSAR Adaptive Platform.			
Base	*ARObject*, *ApmcAbstractDefinition*, *ApmcAbstractInstanceReferenceDef*, *ApmcConfigurationElementDef*, *ApmcContainerElementDef*, *ApmcDefinitionElement*, *ApmcReferenceDef*, *Identifiable*, *MultilanguageReferrable*, *Referrable*			
Aggregated by	ApmcParamConfContainerDef.element			
Attribute	***Type***	***Mult.***	***Kind***	***Note***
–	–	–	–	–

Table 4.28: ApmcInstanceReferenceDef

[3] i.e. derived from meta-class `Referrable`

[AP_TPS_APMC_01024] Semantics of meta-class ApmcUriInstanceReferenceDef ⌈In principle, a ApmcUriInstanceReferenceDef fulfills the same role as a ApmcInstanceReferenceDef.

The difference is that in the case of the ApmcUriInstanceReferenceDef the target of the **structured** reference is **not** required to exist in the same model scope.

On the "value side", if the target of the reference does not exist in the same model scope, then the existence of the ApmcUriInstanceReferenceDef can serve as an **additional identifying element** of the enclosing ApmcContainerValue of the ApmcUriInstanceReferenceDef.⌋

Class	**ApmcUriInstanceReferenceDef**			
Note	This meta-class represents the ability to establish a flat (as opposed to an instanceRef) reference to model elements on design level. In other words, it is possible to annotate which design element was taken as the basis for the target-configuration model element that owns the reference. The target of the reference is not required to exist. The reference itself is already considered an identifying element. This Class is only used by the AUTOSAR Adaptive Platform.			
Base	*ARObject*, *ApmcAbstractDefinition*, *ApmcAbstractInstanceReferenceDef*, *ApmcConfigurationElement Def*, *ApmcContainerElementDef*, *ApmcDefinitionElement*, *ApmcReferenceDef*, *Identifiable*, *MultilanguageReferrable*, *Referrable*			
Aggregated by	ApmcParamConfContainerDef.element			
Attribute	***Type***	***Mult.***	***Kind***	***Note***
–	–	–	–	–

Table 4.29: ApmcUriInstanceReferenceDef

Class	**ApmcUpstreamDocInstanceReferenceDef**			
Note	This meta-class represents the ability to establish a flat (as opposed to an instanceRef) reference to model elements on design level in order to identify a model element that sparked the existence of the owner of the reference. In other words, it is possible to annotate which design element was taken as the basis for the target-configuration model element that owns the reference. This Class is only used by the AUTOSAR Adaptive Platform.			
Base	*ARObject*, *ApmcAbstractDefinition*, *ApmcAbstractInstanceReferenceDef*, *ApmcConfigurationElement Def*, *ApmcContainerElementDef*, *ApmcDefinitionElement*, *ApmcReferenceDef*, *Identifiable*, *MultilanguageReferrable*, *Referrable*			
Aggregated by	ApmcParamConfContainerDef.element			
Attribute	***Type***	***Mult.***	***Kind***	***Note***
–	–	–	–	–

Table 4.30: ApmcUpstreamDocInstanceReferenceDef

[AP_TPS_APMC_01025] Semantics of meta-class ApmcUpstreamDocInstanceReferenceValue ⌈The purpose of the ApmcUpstreamDocInstanceReferenceValue is to establish a **structured** reference to a model **element in the design domain from which the enclosing container was created by derivation**.

The existence of ApmcUpstreamDocInstanceReferenceValue is considered pure metadata.

In contrast to ApmcInstanceReferenceDef and ApmcUriInstanceReferenceDef, the ApmcUpstreamDocInstanceReferenceValue does not carry relevant

model semantics and serves only to indicate how the enclosing container came into existence.⌋

4.2 Value Side

This section contains a detailed description of configuration elements on the “value side”.

4.2.1 Parameter

[AP_TPS_APMC_01026] Semantics of attribute `ApmcIndexableValue.index` ⌈The value of the attribute `ApmcIndexableValue.index` identifies the index of the element in a collection for which the attribute `ApmcConfigurationElement-Def.requiresIndex` has been defined on the “definition side”.⌋

[AP_TPS_APMC_01027] Gaps in the value range of an ordered collection ⌈The values of the individual `index` attributes within an ordered collection may have gaps. The ordering is not based on the usage of every single index value in the affected interval, but on the numerical sorting of the index values.⌋

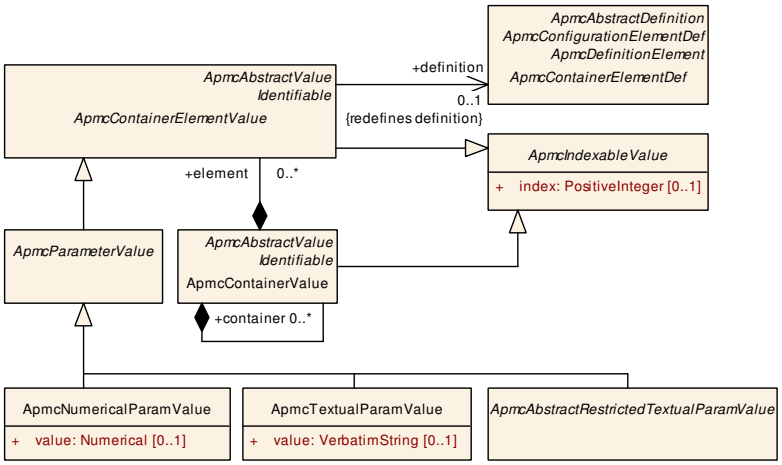


Figure 4.4: Overview of the manifest parameter values

Class	***ApmcIndexableValue*** (abstract)			
Note	This abstract meta-class provides the ability to define model elements that carry an index on the "value" side of the target-configuration.			
Base	*ARObject*			
Subclasses	*ApmcContainerElementValue*, ApmcContainerValue			
Attribute	***Type***	***Mult.***	***Kind***	***Note***

△

Class	ApmcIndexableValue (abstract)			
index	PositiveInteger	0..1	attr	This attribute represents the index value of an element of a collection where the "logical" position of the element in the collection carries a dedicated semantics

Table 4.31: ApmcIndexableValue

4.2.1.1 Numerical Parameter Value

[constr_10584] Allowed `value` for a `ApmcNumericalParamValue` that implements a boolean-style parameter

Imposition time: `IT_CfgFcd`

⌈For any `ApmcNumericalParamValue` that refers to an `ApmcBooleanParamDef` in the role `definition`, the content of the `ApmcNumericalParamValue.value` shall be either 0 or 1.⌋

Class	ApmcNumericalParamValue			
Note	This meta-class represents a numerical parameter on the "value" side of the target-configuration. This Class is only used by the AUTOSAR Adaptive Platform.			
Base	*ARObject*, *ApmcAbstractValue*, *ApmcContainerElementValue*, *ApmcIndexableValue*, *ApmcParameterValue*, *Identifiable*, *MultilanguageReferrable*, *Referrable*			
Aggregated by	ApmcContainerValue.element			
Attribute	**Type**	**Mult.**	**Kind**	**Note**
value	Numerical	0..1	attr	This attribute represents the value of the numerical parameter.

Table 4.32: ApmcNumericalParamValue

[constr_10763] Existence of attribute `ApmcNumericalParamValue.value`

Imposition time: `IT_CfgFcd`

⌈For each `ApmcNumericalParamValue`, the attribute `ApmcNumericalParamValue.value` shall exist.⌋

4.2.1.2 Unrestricted Textual Parameter Value

Because an `ApmcEnumerationParamDef` accepts only defined values, it shall accept only values represented by one of its aggregated literals.

If it is intended to allow for assigning an arbitrary string value to the parameter, an `ApmcStringParamDef` should be preferred.

[constr_10585] Allowed `value` for a `ApmcTextualParamValue` that implements an enumeration-style parameter

Imposition time: IT_CfgFcd

⌈For any ApmcTextualParamValue that refers to an ApmcEnumerationParamDef in the role definition, the content of the ApmcTextualParamValue.value shall only be identical to the shortName of one of the ApmcEnumerationLiteralDefs that are aggregated by the applicable ApmcEnumerationParamDef in the role literal.⌋

Class	ApmcTextualParamValue			
Note	This meta-class represents a textual parameter on the "value" side of the target-configuration. This Class is only used by the AUTOSAR Adaptive Platform.			
Base	*ARObject*, *ApmcAbstractValue*, *ApmcContainerElementValue*, *ApmcIndexableValue*, *ApmcParameterValue*, *Identifiable*, *MultilanguageReferrable*, *Referrable*			
Aggregated by	ApmcContainerValue.element			
Attribute	**Type**	**Mult.**	**Kind**	**Note**
value	VerbatimString	0..1	attr	This attribute represents the value of the textual parameter.

Table 4.33: ApmcTextualParamValue

[constr_10764] Existence of attribute `ApmcTextualParamValue.value`

Imposition time: IT_CfgFcd

⌈For each ApmcTextualParamValue, the attribute ApmcTextualParamValue.value shall exist.⌋

4.2.1.3 Restricted Textual Parameter Value

The definition of restricted textual parameters as a sub-class of ApmcAbstractRestrictedStringParamDef is complemented by the usage of corresponding sub-classes of ApmcAbstractRestrictedTextualParamValue. The modeling of the sub-classes is depicted in Figure 4.5.

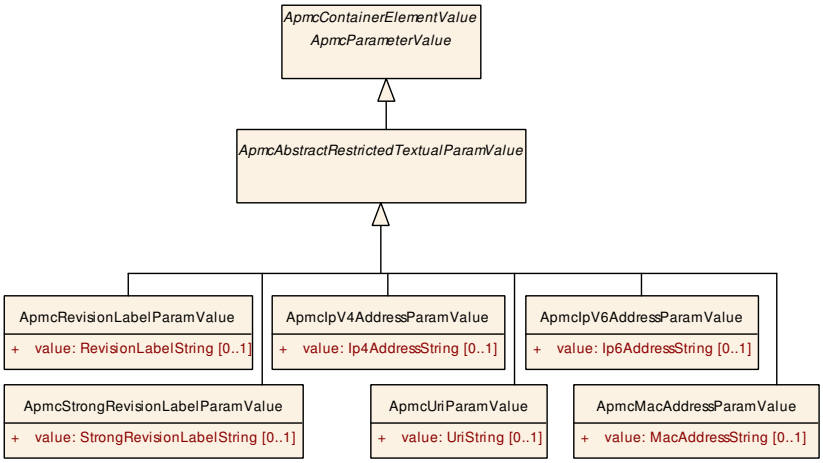


Figure 4.5: Overview of the restricted string parameter values

Each restricted string parameter shall allow the definition of a value that conforms to its applicable restriction. To support this, each subclass defines an attribute that implements the restriction, rather than defining the attribute in the abstract base class.

This modeling is depicted in Figure 4.5.

Of course, it is necessary to make sure that each of the sub-classes of ApmcAbstractRestrictedTextualParamValue refers to the corresponding sub-class of ApmcAbstractRestrictedStringParamDef.

[constr_10635] Allowed target type of the reference in the role ApmcRevisionLabelParamValue.definition

Imposition time: IT_CfgFcd

⌈The reference in the role ApmcRevisionLabelParamValue.definition shall **only** refer to an ApmcRevisionLabelParamDef.⌋

[constr_10636] Allowed target type of the reference in the role ApmcStrongRevisionLabelParamValue.definition

Imposition time: IT_CfgFcd

⌈The reference in the role ApmcStrongRevisionLabelParamValue.definition shall **only** refer to an ApmcStrongRevisionLabelParamDef.⌋

[constr_10637] Allowed target type of the reference in the role ApmcIpV4AddressParamValue.definition

Imposition time: IT_CfgFcd

⌈The reference in the role ApmcIpV4AddressParamValue.definition shall **only** refer to an ApmcIpV4AddressParamDef.⌋

[constr_10638] Allowed target type of the reference in the role ApmcIpV6AddressParamValue.definition

Imposition time: IT_CfgFcd

⌈The reference in the role ApmcIpV6AddressParamValue.definition shall **only** refer to an ApmcIpV6AddressParamDef.⌋

[constr_10639] Allowed target type of the reference in the role ApmcMacAddressParamValue.definition

Imposition time: IT_CfgFcd

⌈The reference in the role ApmcMacAddressParamValue.definition shall **only** refer to an ApmcMacAddressParamDef.⌋

[constr_10640] Allowed target type of the reference in the role ApmcUriParamValue.definition

Imposition time: IT_CfgFcd

⌈The reference in the role ApmcUriParamValue.definition shall **only** refer to an ApmcUriParamDef.⌋

Class	**ApmcAbstractRestrictedTextualParamValue** (abstract)			
Note	This meta-class represents an abstract base class for the value of a textual parameter where the content shall follow a structural restriction on the "definition" side of the target-configuration. The concrete restrictions are defined by sub-classes. This Class is only used by the AUTOSAR Adaptive Platform.			
Base	*ARObject*, *ApmcAbstractValue*, *ApmcContainerElementValue*, *ApmcIndexableValue*, *ApmcParameterValue*, *Identifiable*, *MultilanguageReferrable*, *Referrable*			
Subclasses	ApmcIpV4AddressParamValue, ApmcIpV6AddressParamValue, ApmcMacAddressParamValue, ApmcRevisionLabelParamValue, ApmcStrongRevisionLabelParamValue, ApmcUriParamValue			
Aggregated by	ApmcContainerValue.element			
Attribute	**Type**	**Mult.**	**Kind**	**Note**
–	–	–	–	–

Table 4.34: ApmcAbstractRestrictedTextualParamValue

Class	**ApmcRevisionLabelParamValue**			
Note	This meta-class represents a restricted textual parameter for the specification of a revision label on the "value" side of the target-configuration. This Class is only used by the AUTOSAR Adaptive Platform.			
Base	*ARObject*, *ApmcAbstractRestrictedTextualParamValue*, *ApmcAbstractValue*, *ApmcContainerElementValue*, *ApmcIndexableValue*, *ApmcParameterValue*, *Identifiable*, *MultilanguageReferrable*, *Referrable*			
Aggregated by	ApmcContainerValue.element			
Attribute	**Type**	**Mult.**	**Kind**	**Note**
value	RevisionLabelString	0..1	attr	This attribute implements the actual value of the revision label.

Table 4.35: ApmcRevisionLabelParamValue

[constr_10765] Existence of attribute `ApmcRevisionLabelParamValue.value`

Imposition time: `IT_CfgFcd`

⌈For each `ApmcRevisionLabelParamValue`, the attribute `ApmcRevisionLabelParamValue.value` shall exist.⌋

Class	**ApmcStrongRevisionLabelParamValue**			
Note	This meta-class represents a restricted textual parameter for the specification of a strong revision label on the "value" side of the target-configuration. This Class is only used by the AUTOSAR Adaptive Platform.			
Base	*ARObject*, *ApmcAbstractRestrictedTextualParamValue*, *ApmcAbstractValue*, *ApmcContainerElementValue*, *ApmcIndexableValue*, *ApmcParameterValue*, *Identifiable*, *MultilanguageReferrable*, *Referrable*			
Aggregated by	ApmcContainerValue.element			
Attribute	**Type**	**Mult.**	**Kind**	**Note**
value	StrongRevisionLabelString	0..1	attr	This attribute implements the actual value of the strong revision label.

Table 4.36: ApmcStrongRevisionLabelParamValue

[constr_10766] Existence of attribute ApmcStrongRevisionLabelParamValue.value

Imposition time: IT_CfgFcd

⌈For each ApmcStrongRevisionLabelParamValue, the attribute ApmcStrongRevisionLabelParamValue.value shall exist.⌋

Class	**ApmcIpV4AddressParamValue**			
Note	This meta-class represents a restricted textual parameter for the specification of an IP V4 address on the "value" side of the target-configuration. **Tags:** xml.name=APMC-IP-V4-ADDRESS-PARAM-VALUE This Class is only used by the AUTOSAR Adaptive Platform.			
Base	*ARObject*, *ApmcAbstractRestrictedTextualParamValue*, *ApmcAbstractValue*, *ApmcContainerElement Value*, *ApmcIndexableValue*, *ApmcParameterValue*, *Identifiable*, *MultilanguageReferrable*, *Referrable*			
Aggregated by	ApmcContainerValue.element			
Attribute	***Type***	***Mult.***	***Kind***	***Note***
value	Ip4AddressString	0..1	attr	This attribute implements the actual value of the IP V4 address.

Table 4.37: ApmcIpV4AddressParamValue

[constr_10767] Existence of attribute ApmcIpV4AddressParamValue.value

Imposition time: IT_CfgFcd

⌈For each ApmcIpV4AddressParamValue, the attribute ApmcIpV4AddressParamValue.value shall exist.⌋

Class	**ApmcIpV6AddressParamValue**			
Note	This meta-class represents a restricted textual parameter for the specification of an IP V6 address on the "value" side of the target-configuration. **Tags:** xml.name=APMC-IP-V6-ADDRESS-PARAM-VALUE This Class is only used by the AUTOSAR Adaptive Platform.			
Base	*ARObject*, *ApmcAbstractRestrictedTextualParamValue*, *ApmcAbstractValue*, *ApmcContainerElement Value*, *ApmcIndexableValue*, *ApmcParameterValue*, *Identifiable*, *MultilanguageReferrable*, *Referrable*			
Aggregated by	ApmcContainerValue.element			
Attribute	***Type***	***Mult.***	***Kind***	***Note***
value	Ip6AddressString	0..1	attr	This attribute implements the actual value of the IP V6 address.

Table 4.38: ApmcIpV6AddressParamValue

[constr_10768] Existence of attribute ApmcIpV6AddressParamValue.value

Imposition time: IT_CfgFcd

⌈For each ApmcIpV6AddressParamValue, the attribute ApmcIpV6AddressParamValue.value shall exist.⌋

Class	ApmcMacAddressParamValue			
Note	This meta-class represents a restricted textual parameter for the specification of a MAC address on the "value" side of the target-configuration. This Class is only used by the AUTOSAR Adaptive Platform.			
Base	*ARObject*, *ApmcAbstractRestrictedTextualParamValue*, *ApmcAbstractValue*, *ApmcContainerElement Value*, *ApmcIndexableValue*, *ApmcParameterValue*, *Identifiable*, *MultilanguageReferrable*, *Referrable*			
Aggregated by	ApmcContainerValue.element			
Attribute	**Type**	**Mult.**	**Kind**	**Note**
value	MacAddressString	0..1	attr	This attribute implements the actual value of the MAC address.

Table 4.39: ApmcMacAddressParamValue

[constr_10769] Existence of attribute `ApmcMacAddressParamValue.value`

Imposition time: `IT_CfgFcd`

⌈For each `ApmcMacAddressParamValue`, the attribute `ApmcMacAddressParamValue.value` shall exist.⌋

Class	ApmcUriParamValue			
Note	This meta-class represents a restricted textual parameter for the specification of a URI on the "value" side of the target-configuration. This Class is only used by the AUTOSAR Adaptive Platform.			
Base	*ARObject*, *ApmcAbstractRestrictedTextualParamValue*, *ApmcAbstractValue*, *ApmcContainerElement Value*, *ApmcIndexableValue*, *ApmcParameterValue*, *Identifiable*, *MultilanguageReferrable*, *Referrable*			
Aggregated by	ApmcContainerValue.element			
Attribute	**Type**	**Mult.**	**Kind**	**Note**
value	UriString	0..1	attr	This attribute implements the actual value of the URI.

Table 4.40: ApmcUriParamValue

[constr_10770] Existence of attribute `ApmcUriParamValue.value`

Imposition time: `IT_CfgFcd`

⌈For each `ApmcUriParamValue`, the attribute `ApmcUriParamValue.value` shall exist.⌋

4.2.2 Reference

As depicted in Figure 4.6, all meta-classes that implement references on the “value side” inherit from the abstract base-class `ApmcReferenceValue`.

[AP_TPS_APMC_01037] Semantics of meta-class `ApmcContainerReferenceValue` ⌈Meta-class `ApmcContainerReferenceValue` provides the ability to refer to another `ApmcContainerValue`.

The target is required to exist in the M1 model scope but need not be owned by the same `ApmcFunctionalClusterValue` that also owns the `ApmcContainerValue` in which said `ApmcContainerReferenceValue` is aggregated.⌋

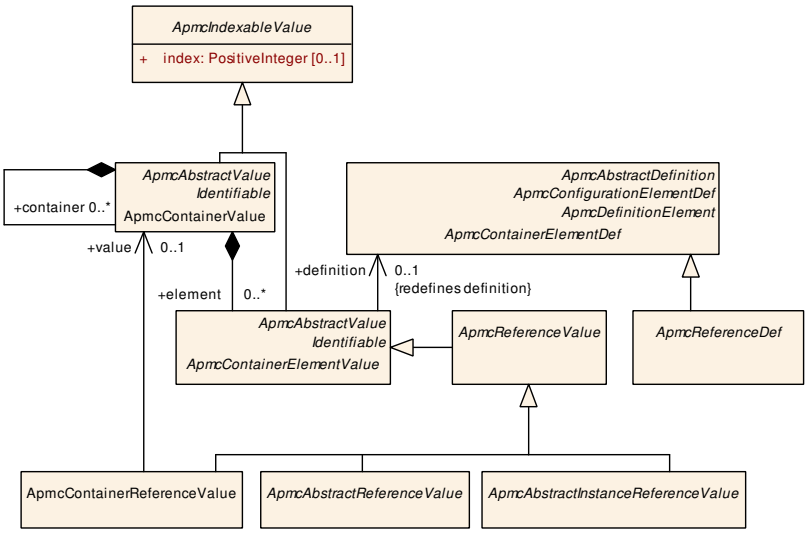


Figure 4.6: Overview of the manifest reference values

[constr_10771] Consistency of `ApmcContainerReferenceValue`

Imposition time: `IT_CfgFcd`

⌈For any given `ApmcContainerReferenceValue`, the `ApmcContainerDef` that is referenced in the role `ApmcContainerReferenceValue.value.definition` shall be identical to the `ApmcContainerDef` referenced in the role `ApmcReference-Value.definition.destination`.⌋

The condition formulated in [constr_10771] is visualized in Figure 4.7.

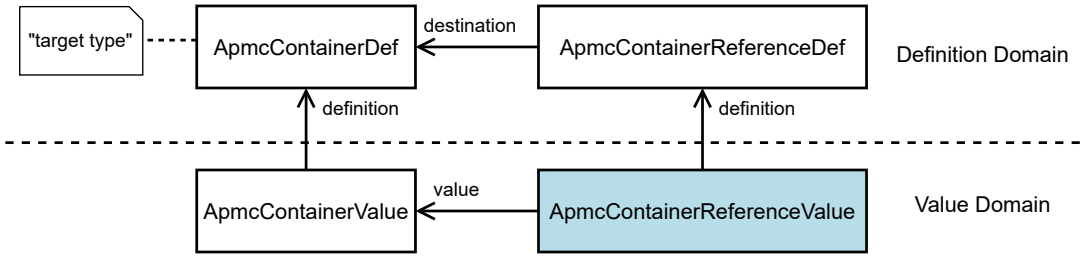


Figure 4.7: Definition of target type for references

Class	**ApmcContainerReferenceValue**			
Note	This meta-class represents a reference to another container of the target-configuration. This Class is only used by the AUTOSAR Adaptive Platform.			
Base	*ARObject*, *ApmcAbstractValue*, *ApmcContainerElementValue*, *ApmcIndexableValue*, *ApmcReference Value*, *Identifiable*, *MultilanguageReferrable*, *Referrable*			
Aggregated by	ApmcContainerValue.element			
Attribute	***Type***	***Mult.***	***Kind***	***Note***
value	ApmcContainerValue	0..1	ref	This referencce identifies the target container. **Tags:** atp.Splitkey=value

Table 4.41: ApmcContainerReferenceValue

[constr_10772] Existence of attribute `ApmcContainerReferenceValue.value`

Status: DRAFT
Imposition time: `IT_CfgFcd`

⌈For each `ApmcContainerReferenceValue`, the attribute `ApmcContainerReferenceValue.value` shall exist.⌋

[AP_TPS_APMC_01028] Semantics of `ApmcAbstractReferenceValue` ⌈Abstract meta-class `ApmcAbstractReferenceValue` is used as a base class for the creation of a **flat** reference to an element of the M2 design model.

In other words, all subclasses of `ApmcAbstractReferenceValue` are used to establish a reference from the configuration model to the design model on the "value side".⌋

The modeling of `ApmcAbstractReferenceValue` and its sub-classes is sketched in Figure 4.8.

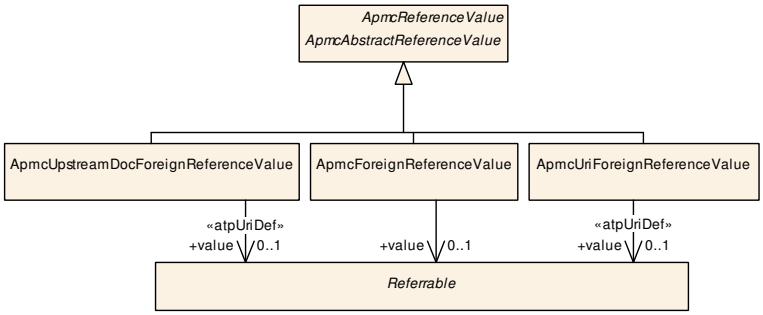


Figure 4.8: Overview of the manifest foreign reference values

Class	***ApmcAbstractReferenceValue*** (abstract)			
Note	This abstract meta-class represents the ability to establish a flat (as opposed to an instanceRef) reference to model elements out of the context of target-configuration. This Class is only used by the AUTOSAR Adaptive Platform.			
Base	*ARObject*, *ApmcAbstractValue*, *ApmcContainerElementValue*, *ApmcIndexableValue*, *ApmcReferenceValue*, *Identifiable*, *MultilanguageReferrable*, *Referrable*			
Subclasses	ApmcForeignReferenceValue, ApmcUpstreamDocForeignReferenceValue, ApmcUriForeignReferenceValue			
Aggregated by	ApmcContainerValue.element			
Attribute	***Type***	***Mult.***	***Kind***	***Note***
–	–	–	–	–

Table 4.42: ApmcAbstractReferenceValue

[AP_TPS_APMC_01029] Semantics of meta-class `ApmcForeignReferenceValue` ⌈Meta-class `ApmcForeignReferenceValue` provides the ability to refer to a referable element of the M2 model by means of the attribute `value`.

The target of this reference is required to exist in the scope of the M2 model.⌋

Class	ApmcForeignReferenceValue			
Note	This meta-class represents the ability to establish a flat (as opposed to an instanceRef) reference to model elements out of the context of target-configuration, i.e. in the design model. This Class is only used by the AUTOSAR Adaptive Platform.			
Base	*ARObject*, *ApmcAbstractReferenceValue*, *ApmcAbstractValue*, *ApmcContainerElementValue*, *ApmcIndexableValue*, *ApmcReferenceValue*, *Identifiable*, *MultilanguageReferrable*, *Referrable*			
Aggregated by	ApmcContainerValue.element			
Attribute	**Type**	**Mult.**	**Kind**	**Note**
value	Referrable	0..1	ref	This reference identifies the target model element that exists outside the target-configuration, i.e. in the design model. **Tags:** atp.Splitkey=value

Table 4.43: ApmcForeignReferenceValue

[constr_10773] Existence of attribute `ApmcForeignReferenceValue.value`

Status: DRAFT
Imposition time: `IT_CfgFcd`

⌈For each `ApmcForeignReferenceValue`, the attribute `ApmcForeignReferenceValue.value` shall exist.⌋

[AP_TPS_APMC_01030] Semantics of meta-class `ApmcUriForeignReferenceValue`

⌈Meta-class `ApmcUriForeignReferenceValue` provides the ability to refer to a referable element of the M2 model by means of the attribute `value`.

The target of this reference is **not** required to exist in the scope of the M2 model. If the target does not exist, then the existence of the reference is taken as an identifying characteristic of the `ApmcContainerValue` that owns the `ApmcUriForeignReferenceValue`.⌋

Class	ApmcUriForeignReferenceValue			
Note	This meta-class represents the ability to establish a flat (as opposed to an instanceRef) reference to model elements out of the context of target-configuration, i.e. in the design model. The referenced model element does not need to exist in the current model scope. The reference itself carries a semantics of being an identifying factor, even without the target element to exist. This Class is only used by the AUTOSAR Adaptive Platform.			
Base	*ARObject*, *ApmcAbstractReferenceValue*, *ApmcAbstractValue*, *ApmcContainerElementValue*, *ApmcIndexableValue*, *ApmcReferenceValue*, *Identifiable*, *MultilanguageReferrable*, *Referrable*			
Aggregated by	ApmcContainerValue.element			
Attribute	**Type**	**Mult.**	**Kind**	**Note**
value	Referrable	0..1	ref	This reference identifies the target model element that exists outside the target-configuration, i.e. in the design model. The referenced model element does not need to exist in the current model scope. The reference itself carries a semantics of being an identifying factor, even without the target element to exist. **Stereotypes:** atpUriDef **Tags:** atp.Splitkey=value

Table 4.44: ApmcUriForeignReferenceValue

[constr_10774] Existence of attribute `ApmcUriForeignReferenceValue.value`

Imposition time: IT_CfgFcd

⌈For each ApmcUriForeignReferenceValue, the attribute ApmcUriForeignReferenceValue.value shall exist.⌋

[AP_TPS_APMC_01031] Semantics of meta-class `ApmcUpstreamDocForeignReferenceValue` ⌈The purpose of the ApmcUpstreamDocForeignReferenceValue is to establish a **flat** reference to a **model element in the design domain from which the enclosing container was created by derivation**.

The existence of ApmcUpstreamDocForeignReferenceValue is considered pure metadata.

In contrast to ApmcForeignReferenceDef and ApmcUriForeignReferenceDef, the ApmcUpstreamDocForeignReferenceValue does not carry relevant model semantics and serves only to indicate how the enclosing container came into existence.⌋

Class	**ApmcUpstreamDocForeignReferenceValue**			
Note	This meta-class represents the ability to establish a flat (as opposed to an instanceRef) reference to model elements out of the context of target-configuration to the model element that sparked the existence of the owner of the reference. This reference does not carry semantics and is meant for documentation purposes only. This Class is only used by the AUTOSAR Adaptive Platform.			
Base	*ARObject*, *ApmcAbstractReferenceValue*, *ApmcAbstractValue*, *ApmcContainerElementValue*, *ApmcIndexableValue*, *ApmcReferenceValue*, *Identifiable*, *MultilanguageReferrable*, *Referrable*			
Aggregated by	ApmcContainerValue.element			
Attribute	***Type***	***Mult.***	***Kind***	***Note***
value	Referrable	0..1	ref	This reference identifies the target model element that exists outside the target-configuration, i.e. in the design model. **Stereotypes:** atpUriDef **Tags:** atp.Splitkey=value

Table 4.45: ApmcUpstreamDocForeignReferenceValue

[constr_10775] Existence of attribute `ApmcUpstreamDocForeignReferenceValue.value`

Imposition time: IT_CfgFcd

⌈For each ApmcUpstreamDocForeignReferenceValue, the attribute ApmcUpstreamDocForeignReferenceValue.value shall exist.⌋

[AP_TPS_APMC_01032] Semantics of `ApmcAbstractInstanceReferenceValue` ⌈Abstract meta-class ApmcAbstractInstanceReferenceValue is used as a base class for the creation of a **structured** reference to an element of the M2 design model.

In other words, all subclasses of ApmcAbstractInstanceReferenceValue are used to establish a reference from the configuration model to the design model on the "value side".⌋

The modeling of ApmcAbstractInstanceReferenceValue and its sub-classes is sketched in Figure 4.9.

Class	***ApmcAbstractInstanceReferenceValue*** (abstract)			
Note	This abstract meta-class represents the ability to establish a instanceRef (as opposed to flat) reference to model elements out of the context of target-configuration. This Class is only used by the AUTOSAR Adaptive Platform.			
Base	*ARObject*, *ApmcAbstractValue*, *ApmcContainerElementValue*, *ApmcIndexableValue*, *ApmcReferenceValue*, *Identifiable*, *MultilanguageReferrable*, *Referrable*			
Subclasses	ApmcInstanceReferenceValue, ApmcUpstreamDocInstanceReferenceValue, ApmcUriInstanceReferenceValue			
Aggregated by	ApmcContainerValue.element			
Attribute	***Type***	***Mult.***	***Kind***	***Note***
–	–	–	–	–

Table 4.46: ApmcAbstractInstanceReferenceValue

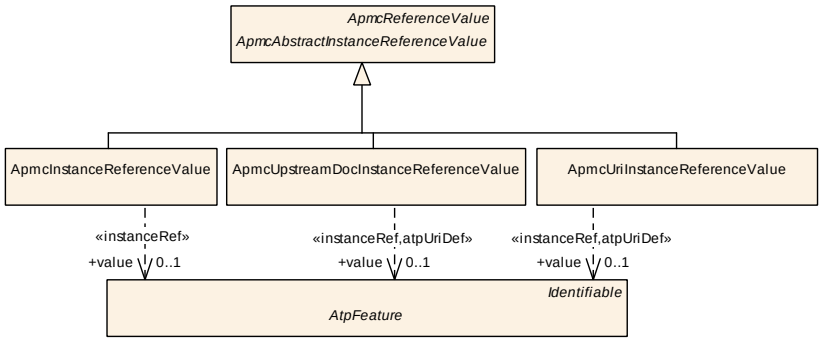


Figure 4.9: Overview of the manifest instance reference values

[AP_TPS_APMC_01033] Semantics of meta-class ApmcInstanceReferenceValue ⌈Meta-class ApmcInstanceReferenceValue provides the ability to establish a **structured** reference to an AtpFeature of the M2 model by means of the attribute value.

The target of this reference is required to exist in the scope of the M2 model.⌋

Class	**ApmcInstanceReferenceValue**			
Note	This meta-class represents the ability to establish a instanceRef (as opposed to flat) reference to model elements out of the context of target-configuration, i.e. in the design model. This Class is only used by the AUTOSAR Adaptive Platform.			
Base	*ARObject*, *ApmcAbstractInstanceReferenceValue*, *ApmcAbstractValue*, *ApmcContainerElementValue*, *ApmcIndexableValue*, *ApmcReferenceValue*, *Identifiable*, *MultilanguageReferrable*, *Referrable*			
Aggregated by	ApmcContainerValue.element			
Attribute	***Type***	***Mult.***	***Kind***	***Note***
value	AtpFeature	0..1	iref	This instanceRef identifies an AtpFeature outside of the target-configuration. **Tags:** atp.Splitkey=value, value.contextElement, value.target **InstanceRef implemented by:** AnyInstanceRef

Table 4.47: ApmcInstanceReferenceValue

[constr_10776] Existence of attribute `ApmcInstanceReferenceValue.value`

Imposition time: IT_CfgFcd

⌈For each ApmcInstanceReferenceValue, the attribute ApmcInstanceReferenceValue.value shall exist.⌋

[AP_TPS_APMC_01034] Semantics of meta-class `ApmcUriInstanceReferenceValue` ⌈Meta-class ApmcUriInstanceReferenceValue provides the ability to establish a **structured** reference to an AtpFeature of the M2 model by means of the attribute value.

The target of this reference is **not** required to exist in the scope of the M2 model. If the target does not exist, then the existence of the reference is taken as an identifying characteristic of the ApmcContainerValue that owns the ApmcUriInstanceReferenceValue.⌋

Class	**ApmcUriInstanceReferenceValue**			
Note	This meta-class represents the ability to establish a instanceRef (as opposed to flat) reference to model elements out of the context of target-configuration, i.e. in the design model. The referenced model element does not need to exist in the current model scope. The reference itself carries a semantics of being an identifying factor, even without the target element to exist. This Class is only used by the AUTOSAR Adaptive Platform.			
Base	*ARObject*, *ApmcAbstractInstanceReferenceValue*, *ApmcAbstractValue*, *ApmcContainerElementValue*, *ApmcIndexableValue*, *ApmcReferenceValue*, *Identifiable*, *MultilanguageReferrable*, *Referrable*			
Aggregated by	ApmcContainerValue.element			
Attribute	***Type***	***Mult.***	***Kind***	***Note***
value	AtpFeature	0..1	iref	This instanceRef identifies an AtpFeature outside of the target-configuration that sparked the existence of the owner of the reference. The referenced model element does not need to exist in the current model scope. The reference itself carries a semantics of being an identifying factor, even without the target element to exist. **Stereotypes:** atpUriDef **Tags:** atp.Splitkey=value, value.contextElement, value.target **InstanceRef implemented by:** AnyInstanceRef

Table 4.48: ApmcUriInstanceReferenceValue

[constr_10777] Existence of attribute `ApmcUriInstanceReferenceValue.value`

Imposition time: IT_CfgFcd

⌈For each ApmcUriInstanceReferenceValue, the attribute ApmcUriInstanceReferenceValue.value shall exist.⌋

[AP_TPS_APMC_01035] Semantics of meta-class `ApmcUpstreamDocInstanceReferenceValue` ⌈The purpose of the ApmcUpstreamDocInstanceReferenceValue is to establish a **structured** reference to a **model element in the design domain from which the enclosing container was created by derivation**.

The existence of ApmcUpstreamDocInstanceReferenceValue is considered pure metadata.

In contrast to ApmcInstanceReferenceDef and ApmcUriInstanceReference-Def, the ApmcUpstreamDocInstanceReferenceValue does not carry relevant model semantics and serves only to indicate how the enclosing container came into existence.⌋

Class	**ApmcUpstreamDocInstanceReferenceValue**			
Note	This meta-class represents the ability to establish a instanceRef (as opposed to flat) reference to model elements out of the context of target-configuration to the model element that sparked the existence of the owner of the reference. This reference does not carry semantics and is meant for documentation purposes only. This Class is only used by the AUTOSAR Adaptive Platform.			
Base	*ARObject*, *ApmcAbstractInstanceReferenceValue*, *ApmcAbstractValue*, *ApmcContainerElementValue*, *ApmcIndexableValue*, *ApmcReferenceValue*, *Identifiable*, *MultilanguageReferrable*, *Referrable*			
Aggregated by	ApmcContainerValue.element			
Attribute	***Type***	***Mult.***	***Kind***	***Note***
value	AtpFeature	0..1	iref	This instanceRef identifies an AtpFeature outside of the target-configuration that sparked the existence of the owner of the reference. **Stereotypes:** atpUriDef **Tags:** atp.Splitkey=value, value.contextElement, value.target **InstanceRef implemented by:** AnyInstanceRef

Table 4.49: ApmcUpstreamDocInstanceReferenceValue

[constr_10778] Existence of attribute ApmcUpstreamDocInstanceReference-Value.value

Imposition time: IT_CfgFcd

⌈For each ApmcUpstreamDocInstanceReferenceValue, the attribute Apmc-UpstreamDocInstanceReferenceValue.value shall exist.⌋

5 Modeling Use Cases

This chapter describes specific modeling use cases in which model elements from this document are combined to form predefined higher-level semantics (from the point of view of the AUTOSAR standard).

5.1 Functional Cluster interacts with Functional Cluster

Functional clusters sometimes need to interact directly with each other. This interaction follows a pattern that resembles, but is not identical to, the “InstanceSpecifier-pattern” for interactions between application software and a functional cluster.

In essence, an `InstanceSpecifier` represents a token known to and exchanged at runtime between two endpoints of an interaction, allowing them to create a “binding.” That is the extent of the analogy between the “InstanceSpecifier-pattern” and the approach described in this chapter.

The token that is known among endpoints of an interaction between two functional clusters has a different structure (i.e. an absolute `shortName` path that belongs to one of the endpoints) than a true `InstanceSpecifier`, which represents the path through an instance tree from the root to a leaf element, typically a `PortPrototype`.

For this reason, and to avoid confusion, the term `InstanceSpecifier` is not part of the pattern for interactions between functional clusters, as described in this chapter.[1]

Instead, the term `FunctionalClusterEndpointSpecifier` is used for this kind of interaction.

For example, the client-side *persistency* functional cluster may request encryption services from the server-side *crypto* functional cluster. This scenario illustrates the modeling approach in Figure 5.1.

[AP_TPS_APMC_01039] Interaction between functional clusters ⌈If an interaction between functional clusters is intended, then the following model structure shall be provided:

- The “client” in the interaction shall define a `ApmcContainerReferenceDef` where attribute `isFuncClusterEndpointRef` is set to `true`.

  The owner of this reference shall represent the configuration of the endpoint for the interaction from the **client**’s point of view.

- The server in the interaction shall define a `ApmcParamConfContainerDef` where attribute `isFuncClusterEndpoint` is set to `true`.

  This container shall represent the configuration of the endpoint for the interaction from the **server**’s point of view.

[1] This statement applies only to the modeling aspect; existing APIs may use the term `InstanceSpecifier` to maintain API stability.

A match between the functional-cluster endpoints is created if and only if

- the value of a ApmcContainerReferenceDef where attribute isFuncClusterEndpointRef is set to true (i.e. the FunctionalClusterEndpointSpecifier) and
- the absolute shortName path of a ApmcParamConfContainerDef where attribute isFuncClusterEndpoint is set to true

are identical.⌋

[constr_10644] Target of a ApmcReferenceValue where definition.isFuncClusterEndpointRef is set to true

Imposition time: IT_CfgFcd

⌈If an ApmcContainerValue is referenced from an ApmcReferenceValue where definition.isFuncClusterEndpointRef is set to true, then the referenced ApmcContainerValue shall refer in the role definition to an ApmcParamConfContainerDef where attribute isFuncClusterEndpoint is also set to true.⌋

The modeling relation described in [AP_TPS_APMC_01039] and [constr_10644] is replicated in the example scenario depicted in Figure 5.1.

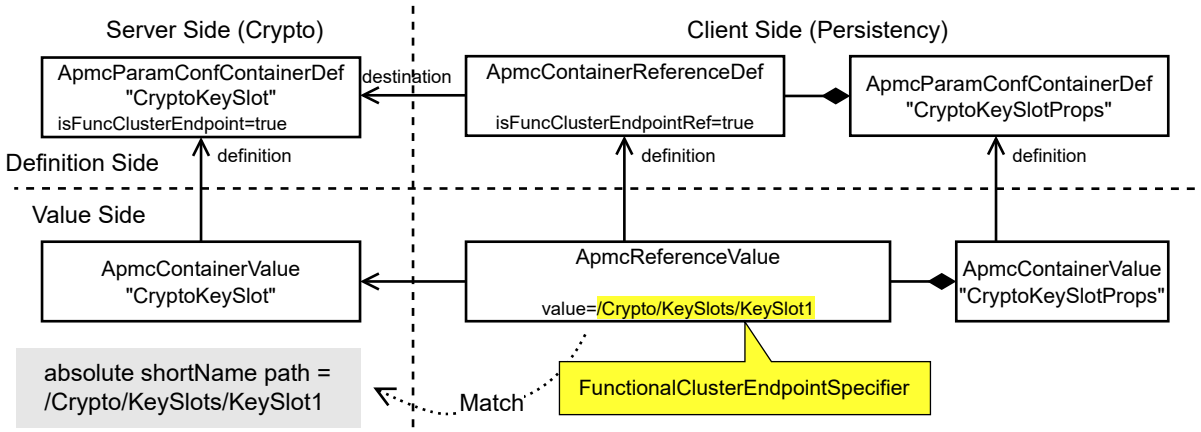


Figure 5.1: Persistency wants to encrypt storage content

A Reference Material

A.1 Terms and Abbreviations

The content of this appendix chapter is *informative* in nature and shall **not** be considered as *normative* content.

The main list of terms and abbreviations are defined in [6]. The following table contains the list of terms and abbreviations used in the scope of this document which are not already defined in [6] along with the spelled-out meaning of each of the abbreviations.

Abbreviation	*Meaning*
APMC	Adaptive Platform Machine Configuration
ARXML	AUTOSAR XML
DM	Diagnostic Manager
ID	Identifier
ISO	International Standardization Organization
JSON	JavaScript Object Notation
NM	Network Management
NV	Non-Volatile
PHM	Platform Health Management
POSIX	Portable Operating System Interface
ROM	Read-Only Memory
SD	Service Discovery
SDG	Special Data Group
SOME/IP	Scalable service-Oriented MiddlewarE over IP
SOVD	Service-Oriented Vehicle Diagnostics
SWC	Software Component
TLV	Tag Length Value
UDS	Unified Diagnostic Services
UML	Unified Modeling Language
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
UUID	Universally Unique Identifier
VSA	Variable Size Array

Table A.1: Terms and Abbreviations used in the scope of this Document

A.2 Imposition Times of Constraints

The constraints formulated in this document have different *actual* imposition times which denote the steps in the workflow when the respective constraint has to be imposed.

The imposition times that are considered applicable in the scope of this document[1] are listed in Table A.2.

Please note that the imposition times are intentionally rendered as technical terms such that it is possible to link back from each constraint to the definition of the affected imposition time in Table A.2.

This document has been created to apply primarily for the *AUTOSAR adaptive platform* and therefore the discussed imposition times also apply exclusively to the *AUTOSAR adaptive platform*.

Imposition Time	Description	Motivation
IT_CfgFcd	Creation of APMC parameter definition model is complete	This imposition time denotes the step in the workflow where the APMC parameter definition of a functional cluster is considered complete. This means that the project development on a given target platform can be started. This imposition time is special in the sense that this happens during AUTOSAR Standard development or Stack Vendor Product development.

Table A.2: Imposition Times of constraints in this document

[1] Different imposition times may be defined in the context of other AUTOSAR standard documents

B Examples

The content of this appendix chapter is *informative* in nature and shall **not** be considered as *normative* content.

There are no examples in this release.

C Change History of AUTOSAR Traceable Items

The content of this appendix chapter is *informative* in nature and shall **not** be considered as *normative* content.

Please note that the lists in this chapter also include constraints and specification items that have been removed from the specification in a later version. These constraints and specification items do not appear as hyperlinks in the document.

C.1 Traceable item history of this document according to AUTOSAR Release R24-11

C.1.1 Added Specification Items in R24-11

Number	Heading
[AP_TPS_APMC_01000]	Abstract relation between the “value side” and the “definition side”
[AP_TPS_APMC_01001]	Root element on the “definition side”
[AP_TPS_APMC_01002]	Semantics of ApmcDefinitionCollection
[AP_TPS_APMC_01003]	Specification of element multiplicity on the “definition side”
[AP_TPS_APMC_01004]	Semantics of attribute ApmcDefinitionElement.elementCreation
[AP_TPS_APMC_01005]	Semantics of attribute ApmcConfigurationElementDef.origin
[AP_TPS_APMC_01006]	Grouping of configuration items on the level of a container
[AP_TPS_APMC_01007]	Semantics of a choice container
[AP_TPS_APMC_01008]	Definition of configuration parameter
[AP_TPS_APMC_01009]	Definition of reference
[AP_TPS_APMC_01010]	Root element on the “value side”
[AP_TPS_APMC_01011]	Definition of an enumeration-style parameter
[AP_TPS_APMC_01012]	Definition of default value for enumeration-style parameter
[AP_TPS_APMC_01013]	Semantics of attribute ApmcConfigurationElementDef.requiresIndex
[AP_TPS_APMC_01014]	Semantics of meta-class ApmcContainerReferenceDef
[AP_TPS_APMC_01015]	Semantics of reference ApmcContainerReferenceDef.destination
[AP_TPS_APMC_01016]	Semantics of meta-class ApmcChoiceContainerReferenceDef
[AP_TPS_APMC_01017]	Semantics of attribute ApmcAbstractForeignReferenceDef.destinationType
[AP_TPS_APMC_01018]	Semantics of ApmcForeignReferenceDef
[AP_TPS_APMC_01019]	Semantics of meta-class ApmcUriForeignReferenceDef
[AP_TPS_APMC_01020]	Semantics of meta-class ApmcUpstreamDocForeignReferenceDef

▽

△

Number	Heading
[AP_TPS_APMC_01021]	Semantics of attribute ApmcAbstractInstanceReferenceDef.destinationContext
[AP_TPS_APMC_01022]	Semantics of attribute ApmcAbstractInstanceReferenceDef.destinationType
[AP_TPS_APMC_01023]	Semantics of ApmcInstanceReferenceDef
[AP_TPS_APMC_01024]	Semantics of meta-class ApmcUriInstanceReferenceDef
[AP_TPS_APMC_01025]	Semantics of meta-class ApmcUpstreamDocInstanceReferenceValue
[AP_TPS_APMC_01026]	Semantics of attribute ApmcIndexableValue.index
[AP_TPS_APMC_01027]	Gaps in the value range of an ordered collection
[AP_TPS_APMC_01028]	Semantics of ApmcAbstractReferenceValue
[AP_TPS_APMC_01029]	Semantics of meta-class ApmcForeignReferenceValue
[AP_TPS_APMC_01030]	Semantics of meta-class ApmcUriForeignReferenceValue
[AP_TPS_APMC_01031]	Semantics of meta-class ApmcUpstreamDocForeignReferenceValue
[AP_TPS_APMC_01032]	Semantics of ApmcAbstractInstanceReferenceValue
[AP_TPS_APMC_01033]	Semantics of meta-class ApmcInstanceReferenceValue
[AP_TPS_APMC_01034]	Semantics of meta-class ApmcUriInstanceReferenceValue
[AP_TPS_APMC_01035]	Semantics of meta-class ApmcUpstreamDocInstanceReferenceValue
[AP_TPS_APMC_01036]	Semantics of ApmcValueCollection
[AP_TPS_APMC_01037]	Semantics of meta-class ApmcContainerReferenceValue
[AP_TPS_APMC_01038]	ApmcFunctionalClusterDef categories

Table C.1: Added Specification Items in R24-11

C.1.2 Changed Specification Items in R24-11

none

C.1.3 Deleted Specification Items in R24-11

none

C.1.4 Added Constraints in R24-11

Number	Heading
[constr_10579]	Existence of `definition`
[constr_10580]	Restriction regarding the value of `ApmcDefinitionElement.lowerMultiplicity` vs. `ApmcDefinitionElement.lowerMultiplicity`
[constr_10581]	Existence of reference in the role `ApmcFunctionalClusterValue.definition`
[constr_10582]	Existence of reference in the role `ApmcContainerValue.definition`
[constr_10583]	Existence of reference in the role `ApmcContainerElementValue.definition`
[constr_10584]	Allowed `value` for a `ApmcNumericalParamValue` that implements an boolean-style parameter
[constr_10585]	Allowed `value` for a `ApmcTextualParamValue` that implements an enumeration-style parameter
[constr_9322]	`ApmcFunctionalClusterDef` category restriction

Table C.2: Added Constraints in R24-11

C.1.5 Changed Constraints in R24-11

none

C.1.6 Deleted Constraints in R24-11

none

C.2 Traceable item history of this document according to AUTOSAR Release R25-11

C.2.1 Added Specification Items in R25-11

Number	Heading
[AP_TPS_APMC_01039]	Interaction between functional clusters

Table C.3: Added Specification Items in R25-11

C.2.2 Changed Specification Items in R25-11

none

C.2.3 Deleted Specification Items in R25-11

none

C.2.4 Added Constraints in R25-11

Number	Heading
[constr_10635]	Allowed target type of the reference in the role `ApmcRevisionLabelParamValue.definition`
[constr_10636]	Allowed target type of the reference in the role `ApmcStrongRevisionLabelParamValue.definition`
[constr_10637]	Allowed target type of the reference in the role `ApmcIpV4AddressParamValue.definition`
[constr_10638]	Allowed target type of the reference in the role `ApmcIpV6AddressParamValue.definition`
[constr_10639]	Allowed target type of the reference in the role `ApmcMacAddressParamValue.definition`
[constr_10640]	Allowed target type of the reference in the role `ApmcUriParamValue.definition`
[constr_10643]	Existence of attributes `ApmcDefinitionElement.upperMultiplicity` and `ApmcDefinitionElement.upperMultiplicityIsInfinite`
[constr_10644]	Target of a `ApmcReferenceValue` where `definition.isFuncClusterEndpointRef` is set to `true`

Table C.4: Added Constraints in R25-11

C.2.5 Changed Constraints in R25-11

none

C.2.6 Deleted Constraints in R25-11

none

C.3 Traceable item history of this document according to AUTOSAR Release R26-11

C.3.1 Added Specification Items in R26-11

Number	Heading
[AP_TPS_APMC_01040]	Matrix of allowed status value combinations of ApmcParamConfContainerDef and aggregations of ApmcParameterDef/ApmcReferenceDef/ApmcContainerDef
[AP_TPS_APMC_01041]	Matrix of allowed status value combinations of referenced targets of a ApmcReferenceDef
[AP_TPS_APMC_01042]	Available choices of ApmcEnumerationParamDefs are defined by aggregated literals
[AP_TPS_APMC_01043]	The shortName of an ApmcEnumerationLiteralDef is used to define the literal
[AP_TPS_APMC_01044]	Definition of interval type for ApmcFloatParamDef.min and max
[AP_TPS_APMC_01045]	Content of attribute ApmcAbstractInstanceReferenceDef.destinationContext
[AP_TPS_APMC_01046]	Content of attribute ApmcAbstractInstanceReferenceDef.destinationType

Table C.5: Added Specification Items in R26-11

C.3.2 Changed Specification Items in R26-11

none

C.3.3 Deleted Specification Items in R26-11

none

C.3.4 Added Constraints in R26-11

Number	Heading
[constr_10756]	Multiplicity of ApmcParamConfContainerDefs aggregated in the role ApmcChoiceContainerDef.choice
[constr_10757]	Existence of the aggregation in the role ApmcEnumerationParamDef.literal
[constr_10758]	Restriction for Attributes ApmcIntegerParamDef.min and max
[constr_10759]	Restriction for Attributes ApmcFloatParamDef.min and max
[constr_10760]	Existence of attribute ApmcAbstractForeignReferenceDef.destinationType

▽

△

Number	Heading
[constr_10761]	Existence of attribute ApmcAbstractInstanceReferenceDef.destinationContext
[constr_10762]	Existence of attribute ApmcAbstractInstanceReferenceDef.destinationType
[constr_10763]	Existence of attribute ApmcNumericalParamValue.value
[constr_10764]	Existence of attribute ApmcTextualParamValue.value
[constr_10765]	Existence of attribute ApmcRevisionLabelParamValue.value
[constr_10766]	Existence of attribute ApmcStrongRevisionLabelParamValue.value
[constr_10767]	Existence of attribute ApmcIpV4AddressParamValue.value
[constr_10768]	Existence of attribute ApmcIpV6AddressParamValue.value
[constr_10769]	Existence of attribute ApmcMacAddressParamValue.value
[constr_10770]	Existence of attribute ApmcUriParamValue.value
[constr_10771]	Consistency of ApmcContainerReferenceValue
[constr_10772]	Existence of attribute ApmcContainerReferenceValue.value
[constr_10773]	Existence of attribute ApmcForeignReferenceValue.value
[constr_10774]	Existence of attribute ApmcUriForeignReferenceValue.value
[constr_10775]	Existence of attribute ApmcUpstreamDocForeignReferenceValue.value
[constr_10776]	Existence of attribute ApmcInstanceReferenceValue.value
[constr_10777]	Existence of attribute ApmcUriInstanceReferenceValue.value
[constr_10778]	Existence of attribute ApmcUpstreamDocInstanceReferenceValue.value

Table C.6: Added Constraints in R26-11

C.3.5 Changed Constraints in R26-11

none

C.3.6 Deleted Constraints in R26-11

none

C.3.7 Added Advisories in R26-11

Number	Heading
[advisory_01022]	Existence of ApmcChoiceContainerDef.choice

Table C.7: Added Advisories in R26-11

C.3.8 Changed Advisories in R26-11

none

C.3.9 Deleted Advisories in R26-11

none

D Mentioned Class Tables

For the sake of completeness, this chapter contains a set of class tables representing meta-classes mentioned in the context of this document but which are not contained directly in the scope of describing specific meta-model semantics.

Class	**AUTOSAR**			
Note	Root element of an AUTOSAR description, also the root element in corresponding XML documents. **Tags:** xml.globalElement=true			
Base	*ARObject*			
Attribute	***Type***	***Mult.***	***Kind***	***Note***
adminData	AdminData	0..1	aggr	This represents the administrative data of an Autosar file. **Tags:** atp.Splitkey=adminData xml.sequenceOffset=10
arPackage	ARPackage	*	aggr	This is the top level package in an AUTOSAR model. **Stereotypes:** atpVariation **Tags:** atp.Splitkey=arPackage, arPackage.shortName, ar Package.variationPoint.shortLabel vh.latestBindingTime=blueprintDerivationTime xml.sequenceOffset=30
fileInfo Comment	FileInfoComment	0..1	aggr	This represents a possibility to provide a structured comment in an AUTOSAR file. **Stereotypes:** atpStructuredComment **Tags:** atp.Splitkey=fileInfoComment xml.roleElement=true xml.sequenceOffset=-10 xml.typeElement=false
introduction	DocumentationBlock	0..1	aggr	This represents an introduction on the Autosar file. It is intended for example to represent disclaimers and legal notes. **Tags:** atp.Splitkey=introduction xml.sequenceOffset=20

Table D.1: AUTOSAR

Class	**AnyInstanceRef**			
Note	Describes a reference to any instance in an AUTOSAR model. This is the most generic form of an instance ref. Refer to the superclass notes for more details.			
Base	*ARObject*, *AtpInstanceRef*			
Aggregated by	ApmcInstanceReferenceValue.value, ApmcUpstreamDocInstanceReferenceValue.value, ApmcUri InstanceReferenceValue.value, Collection.collectedInstance, Collection.sourceInstance, Documentation Context.feature, EcucInstanceReferenceValue.value, FlatInstanceDescriptor.ecuExtractReference, Flat InstanceDescriptor.upstreamReference, RptContainer.byPassPoint, RptHook.rptArHook, SecurityEvent ReportInstanceValue.object, ViewMap.firstElementInstance, ViewMap.secondElementInstance			
Attribute	***Type***	***Mult.***	***Kind***	***Note***
base	AtpClassifier	0..1	ref	This is the base from which navigation path begins. **Stereotypes:** atpDerived
contextElement (ordered)	AtpPrototype	*	ref	This is one step in the navigation path specified by the instance ref.
target	AtpFeature	0..1	ref	This is the target of the instance ref.

Table D.2: AnyInstanceRef

Class	***AtpFeature*** (abstract)			
Note	Features are properties via which a classifier classifies instances. Or: a classifier has features and every M0 instance of it will have those features.			
Base	*ARObject*, *Identifiable*, *MultilanguageReferrable*, *Referrable*			
Subclasses	*AtpPrototype*, *AtpStructureElement*, PhmCheckpoint			
Aggregated by	*AtpClassifier*.atpFeature			
Attribute	***Type***	***Mult.***	***Kind***	***Note***
–	–	–	–	–

Table D.3: AtpFeature

Class	**EcuInstance**			
Note	ECUInstances are used to define the ECUs used in the topology. The type of the ECU is defined by a reference to an ECU specified with the ECU resource description. **Tags:** atp.recommendedPackage=EcuInstances			
Base	*ARObject*, *CollectableElement*, *FibexElement*, *Identifiable*, *MultilanguageReferrable*, *Packageable Element*, *Referrable*			
Aggregated by	ARPackage.element			
Attribute	***Type***	***Mult.***	***Kind***	***Note***
associated ConsumedProvided ServiceInstance Group	ConsumedProvided ServiceInstanceGroup	*	ref	With this reference it is possible to identify which ConsumedProvidedServiceInstanceGroups are applicable for which ECUInstance. **Stereotypes:** atpVariation **Tags:** atp.Splitkey=associatedConsumedProvidedService InstanceGroup, associatedConsumedProvidedService InstanceGroup.consumedProvidedServiceInstanceGroup, associatedConsumedProvidedServiceInstance Group.variationPoint.shortLabel vh.latestBindingTime=postBuild
associatedPdur IPduGroup	PdurIPduGroup	*	ref	With this reference it is possible to identify which PduR IPdu Groups are applicable for which Communication Connector/ ECU. **Tags:** atp.Splitkey=associatedPdurIPduGroup
channel Synchronous Wakeup	Boolean	0..1	attr	If this parameter is available and set to true, then all available channels will be woken up as soon as at least one channel wakeup occurs. If PNCs are configured, then all PNCs will be requested upon a channel wakeup.
clientIdRange	ClientIdRange	0..1	aggr	Restriction of the Client Identifier for this Ecu to an allowed range of numerical values. The Client Identifier of the transaction handle is generated by the client RTE for inter-Ecu Client/Server communication. **Tags:** atp.Splitkey=clientIdRange
commController	Communication Controller	*	aggr	CommunicationControllers of the ECU. **Stereotypes:** atpVariation **Tags:** atp.Splitkey=commController, commController.short Name, commController.variationPoint.shortLabel vh.latestBindingTime=postBuild
connector	Communication Connector	*	aggr	All channels controlled by a single controller. **Stereotypes:** atpVariation **Tags:** atp.Splitkey=connector, connector.shortName, connector.variationPoint.shortLabel vh.latestBindingTime=postBuild

▽

△

Class	**EcuInstance**			
firewallRule	StateDependentFirewall	*	ref	Firewall rules defined in the context of an EcuInstance. **Tags:** atp.Splitkey=firewallRule
pncNmRequest	Boolean	0..1	attr	Defines if this EcuInstance shall request Nm on all its PhysicalChannels which have Nm variant set to FULL each time a PNC is requested.
pncPrepare SleepTimer	TimeValue	0..1	attr	Time in seconds the PNC state machine shall wait in PNC_PREPARE_SLEEP.
pnc Synchronous Wakeup	Boolean	0..1	attr	If this parameter is available and set to true then all available PNCs will be woken up as soon as a channel wakeup occurs. This is ensured by adding all PNCs to all channel wakeup sources during upstream mapping.
pnResetTime	TimeValue	0..1	attr	Specifies the runtime of the reset timer in seconds. This reset time is valid for the reset of PN requests in the EIRA and in the ERA.

Table D.4: EcuInstance

Class	***Identifiable*** (abstract)
Note	Instances of this class can be referred to by their identifier (within the namespace borders). In addition to this, Identifiables are objects which contribute significantly to the overall structure of an AUTOSAR description. In particular, Identifiables might contain Identifiables.
Base	*ARObject*, *MultilanguageReferrable*, *Referrable*
Subclasses	ARPackage, *AbstractDoIpLogicAddressProps*, *AbstractEvent*, *AbstractFunctionalClusterDesign*, *AbstractImplementationDataTypeElement*, *AbstractSecurityEventFilter*, *AbstractSecurityIdsmInstance Filter*, *AbstractServiceInstance*, *AbstractSignalBasedToISignalTriggeringMapping*, AdaptiveSwcInternal Behavior, ApApplicationEndpoint, *ApmcAbstractDefinition*, *ApmcConfigurationElementDef*, *Apmc ContainerElementValue*, ApmcContainerValue, ApmcEnumerationLiteralDef, ApplicationEndpoint, ApplicationError, AppliedStandard, ArtifactChecksum, ArtifactLocator, *AtpBlueprint*, *AtpClassifier*, *Atp Feature*, AutosarOperationArgumentInstance, AutosarVariableInstance, *BuildActionEntity*, BuildAction Environment, CanSecSecureChannel, Chapter, CheckpointTransition, ClientIdDefinition, ClientServer Operation, Code, *CollectableElement*, ComManagementMapping, *CommConnectorPort*, *CommunicationConnector*, *CommunicationController*, Compiler, ConsistencyNeeds, ConsumedEvent Group, CouplingPort, *CouplingPortAbstractShaper*, *CouplingPortStructuralElement*, CryptoCertificate, CryptoCertificateGroup, CryptoKeySlot, CryptoKeySlotDesign, CryptoKeySlotUsageDesign, Crypto Provider, *CryptoServiceMapping*, DataPrototypeGroup, DataPrototypeTransformationPropsIdent, Data Transformation, *DdsAbstractServiceInstanceElementCp*, DdsCpDomain, DdsCpPartition, DdsCpQos Profile, DdsCpTopic, DdsDomainRange, DependencyOnArtifact, *DiagEventDebounceAlgorithm*, *DiagnosticAbstractSovdContent*, DiagnosticAuthTransmitCertificateEvaluation, DiagnosticConnected Indicator, DiagnosticDataElement, DiagnosticDebounceAlgorithmProps, DiagnosticExtendedDataRecord Element, DiagnosticFunctionInhibitSource, DiagnosticParameterElement, *DiagnosticRoutineSubfunction*, DiagnosticSovdMethodPrimitive, DltApplication, DltArgument, DltArgumentProps, DltMessage, DoIp Interface, DoIpLogicAddress, DoIpLogicalAddress, DoIpNetworkConfigurationDesign, DoIpRouting Activation, E2EProfileConfiguration, End2EndEventProtectionProps, End2EndMethodProtectionProps, EthernetWakeupSleepOnDatalineConfig, EventHandler, EventMapping, ExclusiveArea, *Executable Entity*, *ExecutionTime*, FMAttributeDef, FMFeatureMapAssertion, FMFeatureMapCondition, FMFeature MapElement, FMFeatureRelation, FMFeatureRestriction, FMFeatureSelection, FieldMapping, FireAnd ForgetMethodMapping, FlexrayTpPduPool, *FrameTriggering*, GlobalSupervision, GlobalTimeGateway, *GlobalTimeMaster*, *GlobalTimeSlave*, *HealthChannel*, *HeapUsage*, HwAttributeDef, HwAttributeLiteral Def, HwPin, HwPinGroup, *IEEE1722TpAcfBus*, *IEEE1722TpAcfBusPart*, IPSecRule, IPv6ExtHeader FilterList, ISignalToIPduMapping, ISignalTriggering, *IdentCaption*, ImpositionTime, InternalTriggering Point, Keyword, LifeCycleState, Linker, MacAddressVlanMembership, MacMulticastGroup, MacSecKay Participant, McDataInstance, MemorySection, MemoryUsage, MethodMapping, ModeDeclaration, Mode DeclarationMapping, ModeSwitchPoint, ModeSwitchSenderComSpecProps, NetworkEndpoint, *Nm Cluster*, *NmNode*, *PackageableElement*, ParameterAccess, PduActivationRoutingGroup, PduToFrame Mapping, PduTriggering, PerInstanceMemory, *PersistencyDeploymentElement*, *PersistencyInterface Element*, *PhmSupervision*, *PhysicalChannel*, PortGroup, *PortInterfaceMapping*, ProcessToMachine Mapping, Processor, ProcessorCore, PskIdentityToKeySlotMapping, QueuedReceiverComSpecProps, ResourceConsumption, ResourceGroup, RootSwClusterDesignComponentPrototype, RootSw ▽

▽

△

Class	***Identifiable*** (abstract)			
	△ ComponentPrototype, RootSwCompositionPrototype, RptComponent, RptContainer, RptExecutable Entity, RptExecutableEntityEvent, RptExecutionContext, RptProfile, RptServicePoint, RunnableEntity Group, *SdgAttribute*, SdgClass, SecOcJobMapping, SecOcJobRequirement, SecureCommunication AuthenticationProps, *SecureCommunicationDeployment*, SecureCommunicationFreshnessProps, SecurityEventContextDataElement, SecurityEventContextProps, ServerComSpecProps, *Service InterfaceDeploymentElement*, ServiceInterfaceElementSecureComConfig, *ServiceNeeds*, SignalService TranslationEventProps, SignalServiceTranslationProps, SocketAddress, SoftwarePackageStep, Someip EventGroup, SomeipProvidedEventGroup, SomeipTpChannel, *StackUsage*, *StateManagementAction Item*, StateManagementActionList, StateManagementStateNotification, *StateManagementStateRequest*, StaticSocketConnection, StructuredReq, SupervisionCheckpoint, SupervisionMode, SupervisionMode Condition, SwGenericAxisParamType, SwServiceArg, SwcServiceDependency, SystemMapping, *Time BaseResource*, *TimingClock*, TimingClockSyncAccuracy, TimingCondition, *TimingConstraint*, *Timing Description*, TimingExtensionResource, TimingModeInstance, TlsCryptoCipherSuite, TlsCryptoCipher SuiteProps, TlsJobMapping, Topic1, TpAddress, TraceableTable, TraceableText, *TracedFailure*, TransformationISignalPropsIdent, *TransformationProps*, TransformationTechnology, Trigger, Ucm Description, UcmRetryStrategy, UcmStep, UriDescription, VariableAccess, VariationPointProxy, Vehicle RolloutStep, ViewMap, VlanConfig, WaitPoint			
Attribute	***Type***	***Mult.***	***Kind***	***Note***
adminData	AdminData	0..1	aggr	This represents the administrative data for the identifiable object. **Tags:** atp.Splitkey=adminData xml.sequenceOffset=-40
annotation	Annotation	*	aggr	Possibility to provide additional notes while defining a model element (e.g. the ECU Configuration Parameter Values). These are not intended as documentation but are mere design notes. **Tags:** atp.Splitkey=annotation xml.sequenceOffset=-25
category	CategoryString	0..1	attr	The category is a keyword that specializes the semantics of the Identifiable. It affects the expected existence of attributes and the applicability of constraints. **Tags:** xml.sequenceOffset=-50
desc	MultiLanguageOverview Paragraph	0..1	aggr	This represents a general but brief (one paragraph) description what the object in question is about. It is only one paragraph! Desc is intended to be collected into overview tables. This property helps a human reader to identify the object in question. More elaborate documentation, (in particular how the object is built or used) should go to "introduction". **Tags:** atp.Splitkey=desc xml.sequenceOffset=-60
introduction	DocumentationBlock	0..1	aggr	This represents more information about how the object in question is built or is used. Therefore it is a DocumentationBlock. **Tags:** atp.Splitkey=introduction xml.sequenceOffset=-30

▽

△

Class	***Identifiable*** (abstract)			
uuid	String	0..1	attr	The purpose of this attribute is to provide a globally unique identifier for an instance of a meta-class. The values of this attribute should be globally unique strings prefixed by the type of identifier. For example, to include a DCE UUID as defined by The Open Group, the UUID would be preceded by "DCE:". The values of this attribute may be used to support merging of different AUTOSAR models. The form of the UUID (Universally Unique Identifier) is taken from a standard defined by the Open Group (was Open Software Foundation). This standard is widely used, including by Microsoft for COM (GUIDs) and by many companies for DCE, which is based on CORBA. The method for generating these 128-bit IDs is published in the standard and the effectiveness and uniqueness of the IDs is not in practice disputed. If the id namespace is omitted, DCE is assumed. An example is "DCE:2fac1234-31f8-11b4-a222-08002b34c003". The uuid attribute has no semantic meaning for an AUTOSAR model and there is no requirement for AUTOSAR tools to manage the timestamp. **Tags:** xml.attribute=true

Table D.5: Identifiable

Primitive	**Limit**			
Note	This class represents the ability to express a numerical limit. Note that this is in fact a NumericalVariation Point but has the additional attribute intervalType. **Tags:** xml.xsd.customType=LIMIT-VALUE xml.xsd.pattern=(0[xX][0-9a-fA-F]+)|(0[0-7]+)|(0[bB][0-1]+)|(([+\-]?[1-9][0-9]+(\.[0-9]+)?|[+\-]?[0-9](\.[0-9]+)?)([eE]([+\-]?)[0-9]+)?)|\.0|INF|-INF|NaN xml.xsd.type=string			
Attribute	***Type***	***Mult.***	***Kind***	***Note***
intervalType	IntervalTypeEnum	0..1	attr	This specifies the type of the interval. If the attribute is missing the interval shall be considered as "CLOSED". **Tags:** xml.attribute=true

Table D.6: Limit

Class	**Machine**			
Note	Machine that represents an Adaptive Autosar Software Stack. **Tags:** atp.recommendedPackage=Machines This Class is only used by the AUTOSAR Adaptive Platform.			
Base	*ARElement*, *ARObject*, *AtpClassifier*, *AtpFeature*, *AtpStructureElement*, *CollectableElement*, *Identifiable*, *MultilanguageReferrable*, *PackageableElement*, *Referrable*, *UploadableDeployment Element*, *UploadablePackageElement*			
Aggregated by	ARPackage.element, *AtpClassifier*.atpFeature			
Attribute	***Type***	***Mult.***	***Kind***	***Note***
default Application Timeout	EnterExitTimeout	0..1	aggr	This aggregation defines a default timeout in the context of a given Machine with respect to the launching and termination of applications. **Tags:** atp.Splitkey=defaultApplicationTimeout

▽

△

Class	**Machine**			
environment Variable	TagWithOptionalValue	*	aggr	This aggregation represents the collection of environment variables that shall be added to the environment defined on the level of the enclosing Machine. **Tags:** atp.Splitkey=environmentVariable
machineDesign	MachineDesign	0..1	ref	Reference to the MachineDesign this Machine is implementing. **Tags:** atp.Splitkey=machineDesign
module Instantiation	AdaptiveModule Instantiation	*	aggr	Configuration of Adaptive Autosar module instances that are running on the machine. **Tags:** atp.Splitkey=moduleInstantiation, module Instantiation.shortName
processor	Processor	*	aggr	This represents the collection of processors owned by the enclosing machine. **Tags:** atp.Splitkey=processor, processor.shortName
secure Communication Deployment	SecureCommunication Deployment	*	aggr	Target-configuration of secure communication protocol configuration settings to crypto module entities. **Tags:** atp.Splitkey=secureCommunicationDeployment, secure CommunicationDeployment.shortName atp.Status=obsolete
trustedPlatform Executable LaunchBehavior	TrustedPlatform ExecutableLaunch BehaviorEnum	0..1	attr	This attribute controls the behavior of how authentication affects the ability to launch for each Executable.

Table D.7: Machine

Class	***PortPrototype*** (abstract)			
Note	Base class for the ports of an AUTOSAR software component. The aggregation of PortPrototypes is subject to variability with the purpose to support the conditional existence of ports.			
Base	*ARObject*, *AtpFeature*, *AtpPrototype*, *Identifiable*, *MultilanguageReferrable*, *Referrable*			
Subclasses	*AbstractProvidedPortPrototype*, *AbstractRequiredPortPrototype*			
Aggregated by	*AtpClassifier*.atpFeature, *SwComponentType*.port			
Attribute	***Type***	***Mult.***	***Kind***	***Note***
clientServer Annotation	ClientServerAnnotation	*	aggr	Annotation of this `PortPrototype` with respect to client/server communication. **Tags:** atp.Splitkey=clientServerAnnotation
delegatedPort Annotation	DelegatedPort Annotation	0..1	aggr	Annotations on this delegated port. **Tags:** atp.Splitkey=delegatedPortAnnotation
ioHwAbstraction Server Annotation	IoHwAbstractionServer Annotation	*	aggr	Annotations on this IO Hardware Abstraction port. **Tags:** atp.Splitkey=ioHwAbstractionServerAnnotation
modePort Annotation	ModePortAnnotation	*	aggr	Annotations on this mode port. **Tags:** atp.Splitkey=modePortAnnotation
nvDataPort Annotation	NvDataPortAnnotation	*	aggr	Annotations on this non voilatile data port. **Tags:** atp.Splitkey=nvDataPortAnnotation
parameterPort Annotation	ParameterPort Annotation	*	aggr	Annotations on this parameter port. **Tags:** atp.Splitkey=parameterPortAnnotation

▽

△

Class	***PortPrototype*** (abstract)			
portPrototype Props	PortPrototypeProps	0..1	aggr	This attribute allows for the definition of further qualification of the semantics of a PortPrototype. **Tags:** atp.Splitkey=portPrototypeProps This Attribute is only used by the AUTOSAR Adaptive Platform.
senderReceiver Annotation	SenderReceiver Annotation	*	aggr	Collection of annotations of this ports sender/receiver communication. **Tags:** atp.Splitkey=senderReceiverAnnotation
triggerPort Annotation	TriggerPortAnnotation	*	aggr	Annotations on this trigger port. **Tags:** atp.Splitkey=triggerPortAnnotation

Table D.8: PortPrototype

Class	**ProcessDesign**			
Note	This meta-class has the ability to stand in for a Process at the time when the Process does not yet exist. But its future existence already needs to be considered during design phase and for that a dedicated model element is required.. **Tags:** atp.recommendedPackage=ProcessDesigns This Class is only used by the AUTOSAR Adaptive Platform.			
Base	*ARElement*, *ARObject*, *CollectableElement*, *Identifiable*, *MultilanguageReferrable*, *Packageable Element*, *Referrable*			
Aggregated by	ARPackage.element			
Attribute	***Type***	***Mult.***	***Kind***	***Note***
executable	Executable	*	ref	Reference to executable that is executed in the process. **Tags:** atp.Splitkey=executable

Table D.9: ProcessDesign

Class	***Referrable*** (abstract)			
Note	Instances of this class can be referred to by their identifier (while adhering to namespace borders).			
Base	*ARObject*			
Subclasses	*AtpDefinition*, BswDistinguishedPartition, *BswModuleCallPoint*, BswModuleClientServerEntry, BswVariableAccess, CouplingPortTrafficClassAssignment, *CppImplementationDataTypeContextTarget*, *DiagnosticEnvModeElement*, EthernetPriorityRegeneration, ExclusiveAreaNestingOrder, *HwDescriptionEntity*, *ImplementationProps*, ModeTransition, *MultilanguageReferrable*, NmNetworkHandle, PncMappingIdent, *SingleLanguageReferrable*, SoConIPduIdentifier, SomeipRequiredEventGroup, TpConnectionIdent			
Attribute	***Type***	***Mult.***	***Kind***	***Note***
shortName	Identifier	1	attr	This specifies an identifying shortName for the object. It needs to be unique within its context and is intended for humans but even more for technical reference. **Stereotypes:** atpIdentityContributor **Tags:** xml.enforceMinMultiplicity=true xml.sequenceOffset=-100
shortName Fragment	ShortNameFragment	*	aggr	This specifies how the Referrable.shortName is composed of several shortNameFragments. **Tags:** atp.Splitkey=shortNameFragment xml.sequenceOffset=-90

Table D.10: Referrable

Class	**SoftwareCluster**			
Note	This meta-class represents the ability to define an uploadable software-package, i.e. the SoftwareCluster shall contain all software and configuration for a given purpose. **Tags:** atp.recommendedPackage=SoftwareClusters This Class is only used by the AUTOSAR Adaptive Platform.			
Base	*ARElement*, *ARObject*, *CollectableElement*, *Identifiable*, *MultilanguageReferrable*, *Packageable Element*, *Referrable*, *UploadableDeploymentElement*, *UploadablePackageElement*			
Aggregated by	ARPackage.element			
Attribute	***Type***	***Mult.***	***Kind***	***Note***
artifact Checksum	ArtifactChecksum	*	aggr	This aggregation carries the checksums for artifacts contained in the enclosing SoftwareCluster. Please note that the value of these checksums is only applicable at the time of configuration. **Tags:** atp.Splitkey=artifactChecksum, artifact Checksum.shortName
artifactLocator	ArtifactLocator	*	aggr	This aggregation represents the artifact locations that are relevant in the context of the enclosing SoftwareCluster **Tags:** atp.Splitkey=artifactLocator, artifactLocator.short Name
claimed FunctionGroup	ModeDeclarationGroup Prototype	*	ref	Each SoftwareCluster can reserve the usage of a given functionGroup such that no other SoftwareCluster is allowed to use it **Tags:** atp.Splitkey=claimedFunctionGroup
contained ARElement	ARElement	*	ref	This reference represents the collection of model elements that cannot derive from UploadablePackage Element and that contribute to the completeness of the definition of the SoftwareCluster. **Tags:** atp.Splitkey=containedARElement
contained Package Element	UploadablePackage Element	*	ref	This reference identifies model elements that are required to complete the manifest content. **Tags:** atp.Splitkey=containedPackageElement
contained Process	Process	*	ref	This reference represent the processes contained in the enclosing SoftwareCluster. **Tags:** atp.Splitkey=containedProcess
design	SoftwareClusterDesign	*	ref	This reference represents the identification of all Software ClusterDesigns applicable for the enclosing Software Cluster. **Stereotypes:** atpUriDef **Tags:** atp.Splitkey=design
diagnostic Deployment Props	SoftwareCluster DiagnosticDeployment Props	0..1	ref	This reference identifies the applicable SoftwareCluster DiagnosticDeploymentProps that are applicable for the referencing SoftwareCluster. **Tags:** atp.Splitkey=diagnosticDeploymentProps
license	Documentation	*	ref	This attribute allows for the inclusion of the full text of a license of the enclosing SoftwareCluster. In many cases open source licenses require the inclusion of the full license text to any software that is released under the respective license. **Tags:** atp.Splitkey=license
module Instantiation	AdaptiveModule Instantiation	*	ref	This reference identifies AdaptiveModuleInstantiations that need to be included with the SoftwareCluster in order to establish infrastructure required for the installation of the SoftwareCluster. **Tags:** atp.Splitkey=moduleInstantiation

△

Class	**SoftwareCluster**			
releaseNotes	Documentation	0..1	ref	This attribute allows for the explanations of changes since the previous version. The list of changes might require the creation of multiple paragraphs of test. **Tags:** atp.Splitkey=releaseNotes
typeApproval	String	0..1	attr	This attribute carries the homologation information that may be specific for a given country.
vendorId	PositiveInteger	0..1	attr	Vendor ID of this Implementation according to the AUTOSAR vendor list.
vendor Signature	CryptoService Certificate	0..1	ref	This reference identifies the certificate that represents the vendor's signature. **Tags:** atp.Splitkey=vendorSignature
version	StrongRevisionLabel String	0..1	attr	This attribute can be used to describe a version information for the enclosing SoftwareCluster.

Table D.11: SoftwareCluster

Class	**SwComponentPrototype**			
Note	Role of a software component within a composition.			
Base	*ARObject*, *AtpFeature*, *AtpPrototype*, *Identifiable*, *MultilanguageReferrable*, *Referrable*			
Aggregated by	*AtpClassifier*.atpFeature, CompositionSwComponentType.component			
Attribute	*Type*	*Mult.*	*Kind*	*Note*
type	SwComponentType	0..1	tref	Type of the instance. **Stereotypes:** isOfType **Tags:** atp.Splitkey=type

Table D.12: SwComponentPrototype

Class	***Traceable*** (abstract)			
Note	This meta class represents the ability to be subject to tracing within an AUTOSAR model. Note that it is expected that its subclasses inherit either from MultilanguageReferrable or from Identifiable. Nevertheless it also inherits from MultilanguageReferrable in order to provide a common reference target for all Traceables.			
Base	*ARObject*, *MultilanguageReferrable*, *Referrable*			
Subclasses	StructuredReq, *TimingConstraint*, TraceableTable, TraceableText			
Attribute	*Type*	*Mult.*	*Kind*	*Note*
trace	Traceable	*	ref	This association represents the ability to trace to upstream requirements / constraints. This supports for example the bottom up tracing ProjectObjectives <- MainRequirements <- Features <- RequirementSpecs <- BSW/AI **Tags:** atp.Splitkey=trace xml.sequenceOffset=20

Table D.13: Traceable

E Non-Splitable Elements in the Scope of this Document

This chapter contains a table of all non-splitable model elements in the scope of this document.

For more information about the concept of splitable model elements and how these shall be treated please refer to the [5, TPS Generic Structure Template].

Name of non splitable element
ApmcAbstractValue.definition
ApmcEnumerationParamDef.defaultValue

Table E.1: Usage of non splitable elements