Document Title	Technical Report on VSS Representation
Document Category	Technical report (TR)
Document UID	1117
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	• No content changes
2025-11-27	R25-11	AUTOSAR Release Management	• No content changes
2024-11-27	R24-11	AUTOSAR Release Management	• Initial release

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 5
1.1 Objectives . . . 5
2 Methodology 6
2.1 Scope . . . 7
2.1.1 Limitations . . . 7
3 Definition of Terms and Acronyms 8
4 Related Documentation 9
5 VSS Data Catalog Representation 10
5.1 Package Structures . . . 10
5.1.1 Root Package Structures . . . 10
5.1.2 Versioning . . . 11
5.1.3 VSS Branch Package Structure . . . 11
5.2 Primitive Types . . . 12
5.2.1 Numeric Types . . . 12
5.2.2 String Type . . . 12
5.2.2.1 Primitive Sensors, Actuators, and Attributes . . . 13
5.2.2.2 Data Constraints on Primitive Values . . . 14
5.3 Complex Types . . . 16
5.3.1 Arrays . . . 16
5.3.2 Struct Types . . . 18
5.4 Units . . . 19
5.5 Documentation . . . 20
5.6 Custom VSS Key Values . . . 21
6 VISS Service Interfaces Catalog 23
6.1 Service Package Structures . . . 23
6.1.1 Service Package Structures . . . 23
6.2 VISS Service Interfaces . . . 24
6.3 VISS Model Extension . . . 25
6.4 VISS Service Interface Deployments . . . 26
6.5 Provided Service Instance . . . 27
7 Automotive API Gateway 28
7.1 Gateway Software Component . . . 29
A Appendix 31
A.1 Mentioned Manifest Elements . . . 31
A.2 Examples (informative) . . . 56
A.2.1 Root Package Structure . . . 56
A.2.2 Primitive Type Modelling (informative) . . . 58
A.2.3 C++ Keywords . . . 62

A.2.4 Complex Type Modelling 68
A.2.4.1 Variable Size Arrays 68
A.2.4.2 Fixed Size Arrays 70
A.2.4.3 Struct Types 73
A.2.5 VISS Specific Model Extension 77
A.2.6 User Defined VISS Service Deployments 81
A.2.7 Provided Service Instances 83
B Change History of AUTOSAR Traceable Items 84
B.1 Traceable item history of this document according to AUTOSAR Release R26-11 84
B.1.1 Added Specification Items in R26-11 84
B.1.2 Changed Specification Items in R26-11 84
B.1.3 Deleted Specification Items in R26-11 84
B.2 Traceable item history of this document according to AUTOSAR Release R25-11 84
B.2.1 Added Specification Items in R25-11 84
B.2.2 Changed Specification Items in R25-11 84
B.2.3 Deleted Specification Items in R25-11 84
B.3 Traceable item history of this document according to AUTOSAR Release R24-11 85
B.3.1 Added Specification Items in R24-11 85
B.3.2 Changed Specification Items in R24-11 86
B.3.3 Deleted Specification Items in R24-11 87

1 Introduction

AUTOSAR is concerned with providing an automotive base layer catering to automotive application domains.

For an elaborate introduction please consult the explanatory document that covers the reasoning and systematic introduction into the topic.

This document is a technical report how AUTOSAR may be utilized to incorporate the Vehicle Signal Specification into AUTOSAR Applications.

1.1 Objectives

The specific goals are achieved by providing a unified import into AUTOSAR ARXML that is referred to as `VSS Representation`. It thus describes an opinionated view on how a VSS catalog is represented in AUTOSAR. This provides consumers with a unified `VSS Representation` that is beneficial to applications or system components since they have a stable data representation to work with. By incorporating an open standard catalog in AUTOSAR consumers may profit from reduced complexity due to reduced implementation variance across industry while utilizing the existing AUTOSAR technology stack.

See figure 1.1 which outlines the resulting logical model elements.

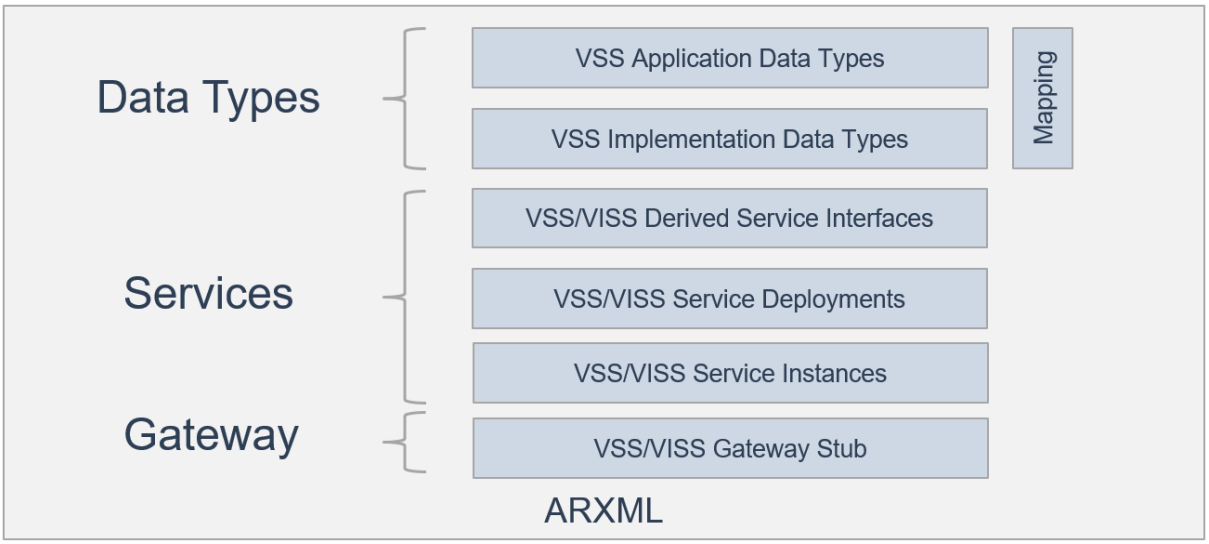


Figure 1.1: VSS Representation Model Elements

The technical report is geared towards the `AUTOSAR Adaptive Platform` but may not be limited to it. `AUTOSAR Classic Platform` may be a viable target as well but is yet to be defined.

2 Methodology

The following figure outlines the methodology how a VSS defined catalog and VSS import to ARXML may be utilized to provide an Automotive API based on ARA::COM.

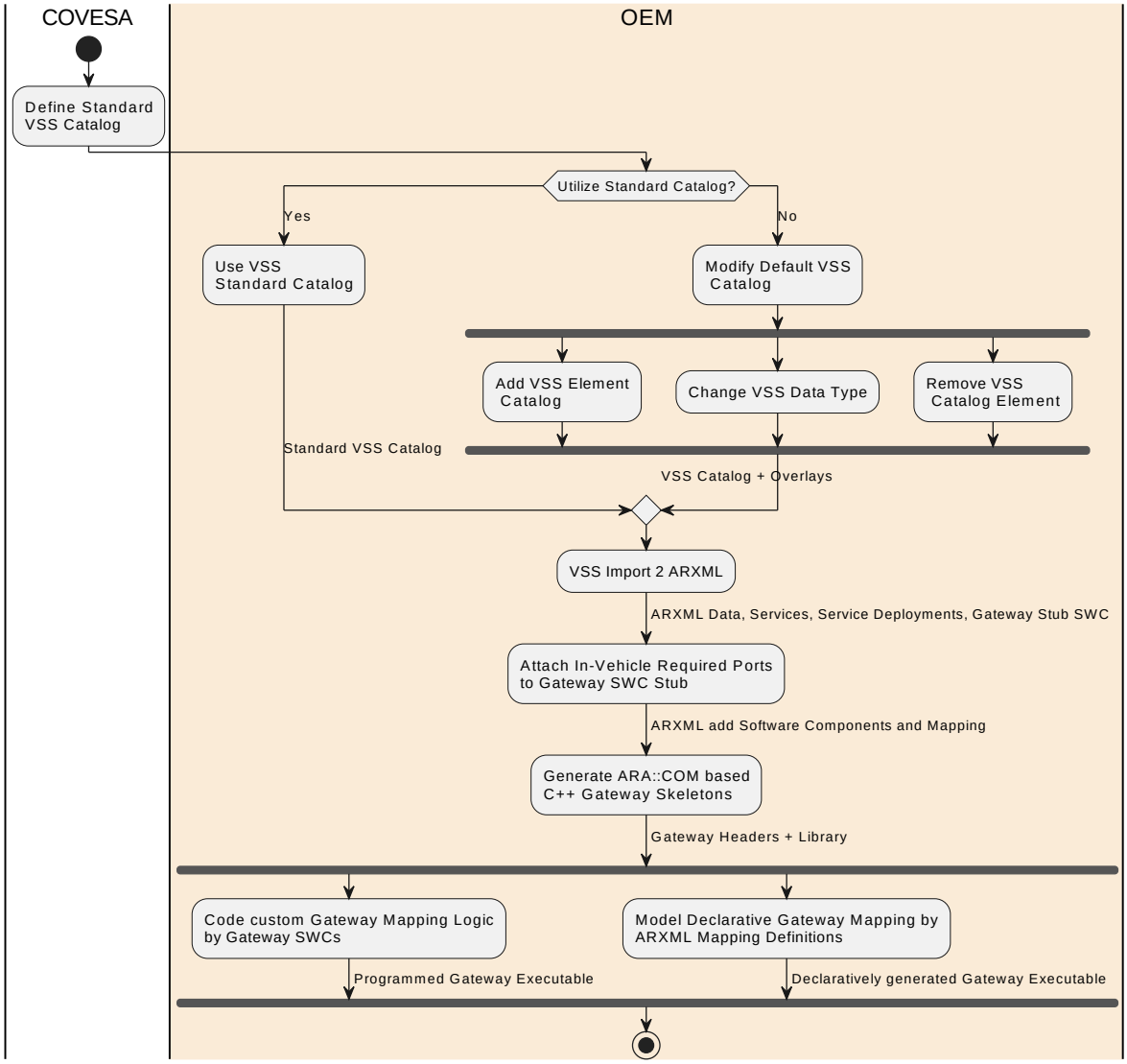


Figure 2.1: VSS based Automotive API Methodology

Starting from the standard VSS catalog an OEM may modify the data catalog and provide additional information that should be provided through the Automotive API. An OEM should be aware that if changes are made in a modifying way, general Automotive API breakage occurs. Removing branches or specific signals may be applicable, if the API Implementation provides runtime errors for absent items.

The VSS importer provides a default service interface definition based on the VSS branch structure. It adopts a specific structure that caters to the VISS semantics and aids in implementing and providing the values correctly.

The defined Automotive API Gateway Stub in 7 allows to have a defined stable endpoint accessible from outside the AUTOSAR ecosystem and maintains the internal flexibility within the AUTOSAR ecosystem to map signals and services by AUTOSAR means.

2.1 Scope

The scope of the document is to provide a unified VSS import. It does not define how a VSS representation is mapped internally to the onboard network as this is subject to OEM specifics.

2.1.1 Limitations

Default values in VSS are currently associated with an instance type. They are currently unsupported by the VSS Representation.

In VSS there is a mechanism to aggregate several nodes to a single branch. As this mechanism is regarded as legacy to the newer struct mechanism, the VSS Representation does not support it.

3 Definition of Terms and Acronyms

Terms:	Description:
VSS [1]	Vehicle Signal Specification defining both a data modelling language and a standard signal catalog
VSS Standard Catalog	The VSS defined standard public catalog containing a set of structured defined vehicle signals
VSS Importer	The entity converting VSS entites to ARXML entities
Vehicle Information Signal Service [2]	An VSS based Automotive API realization over HTTP, MQTT, and Websocket Transports
AUTOSAR Adaptive Platform	see [3] AUTOSAR Glossary
AUTOSAR Classic Platform	see [3] AUTOSAR Glossary
Adaptive Platform Foundation	see [3] AUTOSAR Glossary
Adaptive Application	see [3] AUTOSAR Glossary
VSS Representation	The result of an VSS imporrt described by this document

Table 3.1: Definition of terms in the scope of this Document

4 Related Documentation

[1] Vehicle Signal Specification
https://covesa.github.io/vehicle_signal_specification/

[2] Vehicle Information Service Specification
https://github.com/COVESA/vehicle-information-service-specification/releases/tag/v2.0

[3] Glossary
AUTOSAR_FO_TR_Glossary

5 VSS Data Catalog Representation

In the following sections the following notation is being used to define model elements.

AR Element	This entry defines the AUTOSAR metaclass the VSS element is mapped to. Moreover, a *name* for the target element is introduced in order to refer to the result of the mapping in further entries or rules.
AR Container	This entry specifies the AUTOSAR element that contains the target element defined in the entry above by its *name*.
Attributes	This entry defines the attributes and cross references of the target element.
Condition	In this entry a condition for the mapping can be given. If the condition is false the VSS element does not generate a target element in the AUTOSAR representation.

5.1 Package Structures

5.1.1 Root Package Structures

The recommended AUTOSAR package structure is providing interoperability between AUTOSAR tree structures and the VSS Catalog. It maintains type references that are based on package structure naming.

Refer to the `Identifier` class for naming restrictions of `ARPackage` shortnames.

[AP_TR_VSS_01010] Catalog VSS

Status: DRAFT

⌈

AR Element	`ARPackage` *Catalogs*
AR Container	*AUTOSAR*
Attributes	shortName = "Catalogs"

AR Element	`ARPackage` *VSS*
AR Container	*Catalogs*
Attributes	shortName = "VSS"

AR Element	`ARPackage` *ApplicationDataTypes*
AR Container	*VSS*
Attributes	shortName = "ApplicationDataTypes"

AR Element	`ARPackage` *CppImplementationDataTypes*
AR Container	*VSS*
Attributes	shortName = "CppImplementationDataTypes"

AR Element	`ARPackage` *CompuMethods*
AR Container	*VSS*
Attributes	shortName = "CompuMethods"

AR Element	`ARPackage` *Units*
AR Container	*VSS*
Attributes	shortName = "Units"

⌋

The example A.1 outlines the recommended root package structure shown in ARXML.

5.1.2 Versioning

Versioning must consider the modelling available in the specific VSS version and the VSS standard catalog contents.

Since the catalog version of the standard catalog is a modelling artifact of VSS it is represented in ARXML through mapping. In case support of multiple VSS catalog versions is required the parent package of the catalog may be suffixed with the version number and underscore. As a consequence references to updated catalogs items may be kept stable for updated catalogs and only added or removed items may require updates or references to older catalogs.

[AP_TR_VSS_01018] Package VSS Versions

Status: DRAFT

⌈

AR Element	ARPackage *Version*
AR Container	*VSS*
Attributes	shortName = "Version_4_2"
Condition	—

⌋

The modelling artifacts supported in this technical report are based on VSS version 4.2.

5.1.3 VSS Branch Package Structure

VSS branches are used to logically access vehicle sensors, actuators, or attributes. The VSS Importer preserves that access structure through ARPackages.

VSS Branches in general refer to single physical vehicle entities. For specific cases multiple of the physical entities exist in the vehicle. Examples are tires or doors which may be instantiated multiple times.

In order to honor AUTOSAR Mapping philosophy branch instances shall not be modelled at the type level.

[AP_TR_VSS_01019] VSS Branch Mapping for Types as ARPackage Structure

Status: DRAFT

⌈For each VSS Branch that has no instance property there is an ARPackage. All child VSS Branches are child ARPackages in AUTOSAR. This scheme applies to he ARPackages ApplicationDataTypes, ImplementationDataTypes, and CompuMethods.⌋

The example A.3 showcases the VSS Branch Mapping Structure.

5.2 Primitive Types

The following section outlines the mapping between VSS and AUTOSAR ApplicationPrimitiveDataType and their respective properties and StdCpp-ImplementationDataTypes.

5.2.1 Numeric Types

The numerical base types have no ApplicationPrimitiveDataType representation.

The following table outlines the Mappings to the AUTOSAR Adaptive Platform Implementation Types.

If StdCppImplementationDataTypes are generated by the VSS Importer the platform shall reference StdCppImplementationDataTypes of the standard platform types according to the following table. These types reside in ARPackage /AUTOSAR/StdTypes.

[AP_TR_VSS_02001] VSS Primitive Implementation Data Types

Status: DRAFT

⌈

VSS Data Type:	AUTOSAR StdCppImplementation-DataType:
uint8	uint8_t
uint16	uint16_t
uint32	uint32_t
uint64	uint64_t
int8	int8_t
int16	int16_t
int32	int32_t
int64	int64_t
boolean	bool
float	float
double	double

⌋

5.2.2 String Type

VSS is unicode encoded but leaves open if UTF-8 or UTF-16 is utilized.

[AP_TR_VSS_02003] VSS Primitive String Type

Status: DRAFT

⌈The VSS Importer shall create UTF-8 encoded Strings. The encoding is not modelled. UTF-16 is currently unsupported.⌋

5.2.2.1 Primitive Sensors, Actuators, and Attributes

[AP_TR_VSS_02010] Application Data Type

Status: DRAFT

⌈For each Sensor, Actuator and Attribute that refers to a primitive data type the VSS Importer shall create an ApplicationDataType. It shall be placed into the respective ARPackage that reflects the VSS non instance branch.⌋

[AP_TR_VSS_02011] Application Data Type Name

Status: DRAFT

⌈VSS Importer shall set the ApplicationDataType short name equal to the VSS name.⌋

[AP_TR_VSS_02021] StdCppImplementationDataType

Status: DRAFT

⌈If StdCppImplementationDataTypes are imported for each Sensor, Actuator and Attribute that refers to a primitive data type the VSS Importer shall create an StdCppImplementationDataType. It shall be placed into the respective ARPackage that reflects the VSS non instance branch.⌋

[AP_TR_VSS_02022] StdCppImplementationDataType Name

Status: DRAFT

⌈If StdCppImplementationDataTypes are imported the VSS Importer shall set the StdCppImplementationDataType short name equal to the VSS name.⌋

Importing StdCppImplementationDataType is optional for two reasons. VSS outlines that the implementation representation may be different in the programming language or on the wire from what is defined in the catalog. The default goal of the importer is to reflect the VSS data types as zero cost abstractions in C++ through type aliases. However, it is feasible to resort to already defined AUTOSAR Standard Types and map only those the application data types without the use of type aliases. This will result in less code but may reduce readability for end users. Be aware that applications built on top break in compatibility if StdCppImplementationDataType are modelled with different schemes.

It is therefor optional but recommended that the VSS importer creates StdCppImplementationDataType as separate data types.

[AP_TR_VSS_02023] StdCppImplementationDataType Namespace

Status: DRAFT

⌈VSS namespacing is formed by covesa::vss suffixed by the current VSS type path. The VSS Importer shall create SymbolPropss for each segment of that namespace starting with ns0 as the first name and symbol being the segment. The first name shall count upwards. In case the VSS name represents a C++ Keyword it shall be suffixed with " _"⌋

See A.5 for an example Namespace modelling.

See A.2.3 for the CPP keyword list.

[AP_TR_VSS_02051] DataTypeMappingSet Location

Status: DRAFT

⌈All DataTypeMappingSets shall be contained in the ARPackage Catalog/VSS/-DataTypeMapping.⌋

[AP_TR_VSS_02052] DataTypeMapping

Status: DRAFT

⌈All DataTypeMap may be contained in a single DataTypeMappingSet For each Sensor, Actuator, and Attribute where both an ApplicationDataType and StdCppImplementationDataType has been created the VSS Importer shall create an DataTypeMap which maps the ApplicationDataType to the StdCppImplementationDataType. The DataTypeMap shall be placed into the ARPackage DataTypeMappingSet.⌋

5.2.2.2 Data Constraints on Primitive Values

The AUTOSAR type model allows the definition of Data Constraints. They may further be reused by referencing Data Constraints from multiple AUTOSAR Application Data Types. From the type perspective the constraint is tied to the application data type. To simplify VSS importer implementations data constraint reuse is discouraged for data types that formulate specific data restrictions. Data constraint reuse is allowed only for primitive types.

[AP_TR_VSS_02061] Data Constraint per Data Type

Status: DRAFT

⌈For each ApplicationDataType the VSS Importer shall create a separate PhysConstrs data constraint.⌋

[AP_TR_VSS_02062] Data Constraint Limits

Status: DRAFT

⌈For the VSS node attributes min and max VSS Importer shall create lower and upper limits. The interval type is "CLOSED" for the Lower and Upper limit.⌋

[AP_TR_VSS_02063] Data Constraint Location

Status: DRAFT

⌈The VSS Importer shall create Data Constraints in the same ARPackage as the ApplicationDataType⌋

[AP_TR_VSS_02064] Data Constraint Naming

Status: DRAFT

⌈The VSS Importer shall set the data constraint short name to be identical to the ApplicationDataType with the suffix "DC". If there are multiple data constraints applicable to a single data type, the suffix shall be DCn where n is a natural number starting from 1.⌋

VSS mandates that if no explicit limit of a data type is supplied, the default limits of the referenced primitive data type apply.

For the following primitive types the VSS Importer shall create to following data constraints.

[AP_TR_VSS_02071] VSS Primitive Types Boundary Restrictions

Status: DRAFT

⌈

VSS Data Type:	VSS Min:	VSS Max:	AUTOSAR ADT:	AUTOSAR Adaptive Platform Min:	AUTOSAR Adaptive Platform Max:
uint8	0	255	uint8_t	0	255
uint16	0	65535	uint16_t	0	65535
uint32	0	4294967295	uint32_t	0	4294967295
uint64	0	$2^{(64)} - 1$	uint64_t	0	$2^{(64)} - 1$
int8	-128	127	int8_t	-128	127
int16	-32768	32767	int16_t	-32768	32767
int32	-2147483648	2147483647	int32_t	-2147483648	2147483647
int64	$-2^{(63)}$	$2^{(63)} - 1$	int64_t	$-2^{(63)}$	$2^{(63)} - 1$
boolean	0/false	1/true	bool	0/false	1/true
float	$-3.4e-38$	$3.4e+38$	float	$-3.402823466e+38$	$3.402823466e+38$
double	$-1.80e+308$	$1.80e+308$	double	$-1.80e+308$	$1.80e+308$

⌋

See A.6 for an example Data Constraint modelling.

[AP_TR_VSS_02072] VSS String Enum Restrictions

Status: DRAFT

⌈For each VSS leaf that is of primitive value with allowed values the VSS Importer shall produce a CompuMethod contaning a TEXTTABLE with the allowed VSS values. The VSS Importer shall set the name of the CompuMethod to the VSS data type name suffixed with CM. The VSS Importer shall create a CompuScale for an enumeration element. The VSS Importer shall create a LowerLimit and an UpperLimit for each enumeration element item. The VSS Importer shall set the value to the current element count. The VSS Importer shall create a CompuConst setting the actual allowed string value of the VSS enumeration element. The VSS Importer shall reference the produced CompuMethod through SwDataDefProps.⌋

Additional enumeration comments clarifying the semantics are not respected in AUTOSAR

See A.7 for an example of enumeration restriction.

5.3 Complex Types

5.3.1 Arrays

[AP_TR_VSS_03101] Application Array Data Type

Status: DRAFT

⌈For each VSS node of type Sensor, Actuator and Attribute that has an array type the VSS Importer shall create an ApplicationArrayDataType with its category is set to ARRAY.⌋

[AP_TR_VSS_03104] Application Array Name

Status: DRAFT

⌈The VSS Importer shall set ApplicationArrayDataType short name to the VSS node name suffixed with "Array".⌋

[AP_TR_VSS_03105] Application Array Element Data Type

Status: DRAFT

⌈For each VSS node of type Sensor, Actuator and Attribute that has an array type the VSS Importer shall create an ApplicationDataType.⌋

[AP_TR_VSS_03106] Application Array Element Name

Status: DRAFT

⌈The VSS Importer shall set ApplicationDataType short name to the VSS node name.⌋

[AP_TR_VSS_03102] Application Array Data Type Reference

Status: DRAFT

⌈The VSS Importer shall create a type reference from the array to the corresponding ApplicationDataType element. This may be a ApplicationPrimitiveDataType, an ApplicationArrayDataType or an ApplicationRecordDataType.⌋

[AP_TR_VSS_03103] Application Array Size Handling

Status: DRAFT

⌈The VSS Importer shall set the property ArraySizeSemantics to

FIXED-SIZE

if the VSS leaf defines the arraysize property and

VARIABLE-SIZE

if the arraysize property is undefined. The VSS Importer shall create an element maxNumberOfElements as PositiveIntegerValueVariationPoint and write the contents of arraysize to the BlueprintValue.⌋

See A.8 for an example application array data type with variable size.

[AP_TR_VSS_03110] C++ Implementation Array Data Type

Status: DRAFT

⌈If StdCppImplementationDataTypes are imported then for each Sensor, Actuator and Attribute the VSS Importer shall create a StdCppImplementationDataType. If the VSS node has the attribute arraySize set, the category shall be set to "ARRAY". If the VSS node has no attribute arraySize set, the category shall be set to "VECTOR".⌋

[AP_TR_VSS_03111] C++ Implementation Array Data Reference

Status: DRAFT

⌈If StdCppImplementationDataTypes are imported the VSS Importer shall create a CppTemplateArgument that references the corresponding StdCppImplementationDataType.⌋

[AP_TR_VSS_03112] C++ Implementation Array Name

Status: DRAFT

⌈If StdCppImplementationDataTypes are imported the VSS Importer shall set the StdCppImplementationDataType short name that refers to the array to the VSS node name suffixed with "Array".⌋

See A.9 for an example array implementation data type.

5.3.2 Struct Types

[AP_TR_VSS_03201] Application Record Data Type

Status: DRAFT

⌈For each VSS node of type struct VSS Importer shall create an ApplicationRecordDataType. The category is set to STRUCTURE.⌋

[AP_TR_VSS_03202] Application Record Name

Status: DRAFT

⌈The VSS Importer shall set ApplicationRecordDataType shortname to the VSS node name.⌋

[AP_TR_VSS_03203] Application Record Element Type

Status: DRAFT

⌈For each VSS node of type struct VSS Importer shall create an ApplicationRecordElement. The category is set to "VALUE".⌋

[AP_TR_VSS_03204] Application Record Element Type Reference

Status: DRAFT

⌈The VSS Importer shall resolve VSS primitive data type properties to ApplicationPrimitiveDataTypes in the type reference. The VSS Importer shall reference properties of the VSS struct data type to the correspondig ApplicationRecordDataType in the type reference.⌋

See example A.12 for modelling of structs with ApplicationDataTypes.

[AP_TR_VSS_03211] C++ Implementation Struct Data Type

Status: DRAFT

⌈If StdCppImplementationDataTypes are imported the VSS Importer shall create a StdCppImplementationDataType. The category is set to "STRUCTURE".⌋

[AP_TR_VSS_03212] C++ Implementation Element Type Reference

Status: DRAFT

⌈If StdCppImplementationDataTypes are imported the VSS Importer shall resolve VSS primitive data types and struct data type properties to StdCppImplementationDataType in the type reference.⌋

[AP_TR_VSS_03213] C++ Implementation Struct Array Name

Status: DRAFT

⌈The VSS Importer shall set the short name to the VSS node property name of the struct.⌋

[AP_TR_VSS_03214] C++ Implementation Element Array Name

Status: DRAFT

⌈The VSS Importer shall set the short name to the VSS node property name.⌋

See example A.13 for modelling of structs with StdCppImplementationDataTypes.

5.4 Units

Units define standards for expressing and comparing measures relating to physical quantities.

VSS defines a set of units within specified unit files which abbreviations open for extension and modification. AUTOSAR on its behalf defines blueprint for Units.

Both specifications have Units in common but also define additional Units.

VSS mentions that the default catalog defines default units, but APIs may deviate from that setting.

AUTOSAR supports that philosophy through ApplicationDataTypes. ApplicationDataTypes reference units. CompuMethods convert values from ApplicationDataType to StdCppImplementationDataTypes and vice versa on the modelling level. This may convert a speed available in km/h in VSS and on AUTOSAR ApplicationDataType to m/s on StdCppImplementationDataType.

[AP_TR_VSS_04401] Common Unit Mapping

Status: DRAFT

⌈For VSS Units that equal semantically the AUTOSAR counterpart the VSS Importer shall not import the respective units.⌋

Example of matching VSS and AUTOSAR Units are

- mm
- km
- km/h
- l

[AP_TR_VSS_02402] VSS Units

Status: DRAFT

⌈VSS Units that have no representation in AUTOSAR shall be imported.⌋

Example of Units that are defined in AUTOSAR but not in VSS

- cm
- inch

- lbs
- kWh/100km

[AP_TR_VSS_04003] VSS Units Package

Status: DRAFT

⌈Absent AUTOSAR Units that are present in VSS shall be imported by the VSS Importer to the VSS Units Package.⌋

VSS quantities that relate units to be compatible are not considered in AUTOSAR.

[AP_TR_VSS_04404] VSS Units Reference

Status: DRAFT

⌈VSS Units shall be referenced by the ApplicationDataType through SwDataDefProps.⌋

5.5 Documentation

For each primitive, array, struct and struct element VSS node documentation the VSS Importer shall create respective documentation structures in ApplicationPrimitiveDataType ApplicationArrayDataType ApplicationRecordDataType ApplicationRecordElement The VSS Importer shall create a MultiLanguageOverviewParagraph. The VSS Importer shall create a LOverviewParagraph. The VSS Importer shall create a MixedContentForOverviewParagraph for the VSS description. If there is no comment in VSS a MixedContentForOverviewParagraph shall not be created. The VSS Importer shall create a DocumentationBlock for the VSS comment. If there is a comment in the VSS node, the VSS Importer shall create a Note for the VSS comment. The contents of the comment shall be set by the the VSS Importer to the contents of the VSS node comment.

[AP_TR_VSS_04440] VSS Description Base Structure

Status: DRAFT

⌈

AR Element	MultiLanguageOverviewParagraph
AR Container	*ApplicationDataType*

⌋

[AP_TR_VSS_04441] VSS Description Base Structure

Status: DRAFT

⌈

AR Element	LOverviewParagraph
AR Container	*MultiLanguageOverviewParagraph*

⌋

[AP_TR_VSS_04442] VSS Description

Status: DRAFT

⌈

AR Element	MixedContentForOverviewParagraph
AR Container	*LOverviewParagraph*

⌋

[AP_TR_VSS_04443] VSS Comment Base Structure

Status: DRAFT

⌈

AR Element	DocumentationBlock
AR Container	*ApplicationDataType*

⌋

[AP_TR_VSS_04444] VSS Comment

Status: DRAFT

⌈

AR Element	Note
AR Container	*DocumentationBlock*

⌋

5.6 Custom VSS Key Values

VSS allows the definition of custom key value pairs through overlays for nodes. It is desirable to maintain that information in AUTOSAR ARXML since it may encode relevant information.

[AP_TR_VSS_04450] VSS Key and Value Base Structure

Status: DRAFT

⌈If a VSS node contains at least one key value pair, the VSS Importer shall create an AdminData model element on the ApplicationDataType representing the VSS node. The VSS Importer shall create a Special Data Group Sdg setting the gid attribute to VSS.⌋

[AP_TR_VSS_04451] VSS Key and Value modelling

Status: DRAFT

⌈For each overlay key and value pair the `VSS Importer` shall create a `Sd`. The `VSS Importer` shall set the gid attribute to the `VSS` node overlay key and the value attribute to the `VSS` node overlay value.⌋

The example A.14 outlines a key value pair representation in ARXML.

6 VISS Service Interfaces Catalog

6.1 Service Package Structures

6.1.1 Service Package Structures

The recommended AUTOSAR package structure is as follows.

Refer to the `Identifier` class for naming restrictions of `ARPackage` short names.

[AP_TR_VSS_05011] AutomotiveAPI

Status: DRAFT

⌈

AR Element	`ARPackage` *AutomotiveAPI*
AR Container	*AUTOSAR*
Attributes	shortName = "AutomotiveAPI"
AR Element	`ARPackage` *VISS*
AR Container	*AutomotiveAPI*
Attributes	shortName = "VISS"
AR Element	`ARPackage` *ServiceInterfaces*
AR Container	*VISS*
Attributes	shortName = "ServiceInterfaces"
AR Element	`ARPackage` *ServiceInterfaces*
AR Container	*VISS*
Attributes	shortName = "ServiceInterfaces"
AR Element	`ARPackage` *AutomotiveAPIGateway*
AR Container	*AutomotiveAPI*
Attributes	shortName = "Gateway"
AR Element	`ARPackage` *ServiceInstances*
AR Container	*Gateway*
Attributes	shortName = "ServiceInstances"
AR Element	`ARPackage` *ServiceInstanceMappings*
AR Container	*Gateway*
Attributes	shortName = "ServiceInstanceMappings"
AR Element	`ARPackage` *SWC*
AR Container	*Gateway*
Attributes	shortName = "SWC"

⌋

VSS does not define service interfaces. The tree formed by branches provides logical grouping that can be utilized to form service interfaces. The way that VSS is modelled this provides a convenient logical structure for application consumers. On the other hand utilizing this structure may not be well applicable for service providers since only portions of the contained fields, methods or events are available at the deployment location. In such a scenario flexibility in separating or aggregating VSS sensor and actuators is required. This is a problem if a unified In-Vehicle API based on ARA::COM

or SOME/IP shall be provided. Consumers of an API then require deployment specific information reducing the non-functional requirement of portability with respect to consumers of such an API.

The VSS specification does not specify an application API but suggests that Vehicle APIs based on VSS operate in a declarative style.

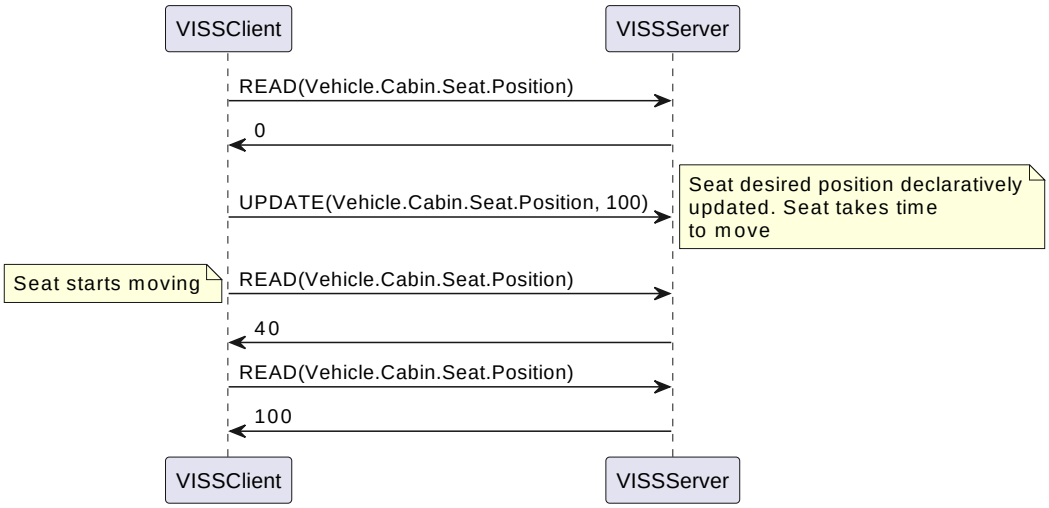


Figure 6.1: Declarative API Example

See figure 6.1 for an example seat position update through VISS.

In a declarative style API there is a distinction between a desired value to be taken and the current value of the entity. By utilizing such an API style the supervision of reaching the desired state remains close to the sensors and actuators and clients can be strongly decoupled from remote sensors and actuators because they have less responsibility. A downside is that for clients that realize remotely driven control loops the declarative API style is unfavorable.

Applicability of such an API style onboard the vehicle may be limited due to resource constraints since two distinct values would be required to be stored in memory. Another limiting factor to applicability may be the scenario of co-design of client and server which form a coherently behaving control loop.

6.2 VISS Service Interfaces

The goal is therefore to provide an API that realizes a Declarative Service Interface to fulfill the automotive gateway use case. Other use cases may or may not make use of this API style depending on their needs.

ARA::COM does not provide a declarative semantic in a single access primitive. However by a combination of several ARA::COM access primitives the declarative behavior can be established. Utilizing more than one access primitive yields the benefit to be

more explicit when mapping primitives to in vehicle accesses. A drawback is that more authoring is required as a consequence.

In order to realize the declarative semantics a field is utilized for data getting and notification. A method is used for setting the data value declaratively.

[AP_TR_VSS_05201] Service Interfaces

Status: DRAFT

⌈The `VSS Importer` shall produce a `ServiceInterface` for each `VSS` branch that is not an instance.⌋

Be aware that `VSS` aggregate:true does not have semantics in the VSS Representation and shall not produce a struct of the contained elements.

[AP_TR_VSS_05202] Data Retrieval

Status: DRAFT

⌈The `VSS Importer` shall produce a `Field` for each VSS Sensor, Actuator and Attribute. The field shall have hasGetter and hasNotifier attributes set to true while having the hasSetter attribute set to false. The field short name shall reflect the last name of the VSS sensor, attribute or actuator. The `VSS Importer` shall create the `Field` in the respective `ServiceInterface`.⌋

[AP_TR_VSS_05203] Data Updates

Status: DRAFT

⌈The `VSS Importer` shall produce a `ClientServerOperation` with the VSS node data type as single argument to be set. The method short name shall reflect the last name of the VSS sensor, attribute or actuator prefixed by set. The `VSS Importer` shall create the `ClientServerOperation` in the respective `ServiceInterface`.⌋

6.3 VISS Model Extension

To model the VISS communication UserDefined model artifacts are utilized. In order to distinguish between other UserDefined model artifacts a qualification that the UserDefined model artifacts are indeed VISS is required.

See example A.15 for an example of a model extension for VISS.

This model extension is utilized in the following sections to create the necessary artifacts.

6.4 VISS Service Interface Deployments

[AP_TR_VSS_05210] VISS/VSS Service Deployments

Status: DRAFT

⌈For each VSS based Service the VSS Importer shall create an UserDefinedServiceInterfaceDeployment. It shall reference the VSS associated ServiceInterface through a ServiceInterfaceARRef⌋

[AP_TR_VSS_05215] User Defined VISS Service Instance

Status: DRAFT

⌈For each UserDefinedServiceInterfaceDeployment that reflects a VSS derived service interface the VSS Importer shall create an AdminData. Each AdminData shall contain a Special Data Group Sdg with the gid suffixed by ProvidedVISSBinindServiceInstanceExtension. The SpecialDataGroup shall contain a Special Data Group Sdg with Gid set to userDefinedVISSBinding. This in turn shall contain a Special Data Group Sdg with Gid set to ProvidedVISSBindingServiceInstance. This in turn shall contain a Special Data Sd with Gid vssInstancePath and the value set to the VSS instance path in dot notation.⌋

See example A.16 for an example of VISS specific UserDefinedServiceInterfaceDeployments.

[AP_TR_VSS_05211] VISS/VSS Field Deployments

Status: DRAFT

⌈For each VSS based Service Field the VSS Importer shall create an UserDefinedFieldDeployment that references the associated field.⌋

[AP_TR_VSS_05212] VISS/VSS Set Method Deployments

Status: DRAFT

⌈For each VSS based Service setter method the VSS Importer shall create an UserDefinedMethodDeployment that references the associated set method. The short name shall follow the pattern set<Shortname>(<Datatype>).⌋

[AP_TR_VSS_05213] VISS/VSS Field Get Deployments

Status: DRAFT

⌈For each UserDefinedFieldDeployment the VSS Importer shall create a UserDefinedMethodDeployment for a get which shall have the short name get<FieldName>.⌋

[AP_TR_VSS_05214] VISS/VSS Field Notifier Deployments

Status: DRAFT

⌈For each UserDefinedFieldDeployment the VSS Importer shall create a UserDefinedEventDeployment for set which shall have the short name on<FieldName>.⌋

6.5 Provided Service Instance

[AP_TR_VSS_05220] Provided Service Instance Package Structure

Status: DRAFT

⌈For each VSS Branch the VSS Importer shall create an ARPackage. All child VSS Branches are child ARPackages in AUTOSAR. The modelling of ProvidedVISSServiceInstance packages differs from the data type modelling since instances of branches are modelled as additional ARPackages.⌋

[AP_TR_VSS_05221] Provided Service Instance

Status: DRAFT

⌈For each VSS node instance that reflects a VSS derived service interface the VSS Importer shall produce a ProvidedUserDefinedServiceInstance. The short name shall be equal to the service name. The ProvidedUserDefinedServiceInstance parent ARPackage shall be set by the VSS Importer to the branch instance ARPackage. The VSS Importer shall reference the corresponding VISS service interface deployment through the serviceInterfaceDeployment property.⌋

See example A.17 for an example of VISS specific UserDefinedServiceInterfaceDeployments.

[AP_TR_VSS_05222] Provided Service Instance VSS Path

Status: DRAFT

⌈The VSS Importer shall set the vssInstancePath property of ProvidedUserDefinedServiceInstance of the model extension to the fully qualified VSS instance path.⌋

[AP_TR_VSS_05223] Provided Service Instance Deployment Reference

Status: DRAFT

⌈The VSS Importer shall set the serviceInterfaceDeployment reference of ProvidedUserDefinedServiceInstance to the UserDefinedServiceInterfaceDeployment.⌋

7 Automotive API Gateway

This section outlines the modelling portions of the Automotive API Gateway.

The goal is to provide a well-defined framework and abstraction on the application layer for mapping standard defined VSS/VISS interfaces and application data types to OEM provided vehicle internal interfaces and application data types. Its purpose is to hide transport implementation details in order to ease the mapping steps.

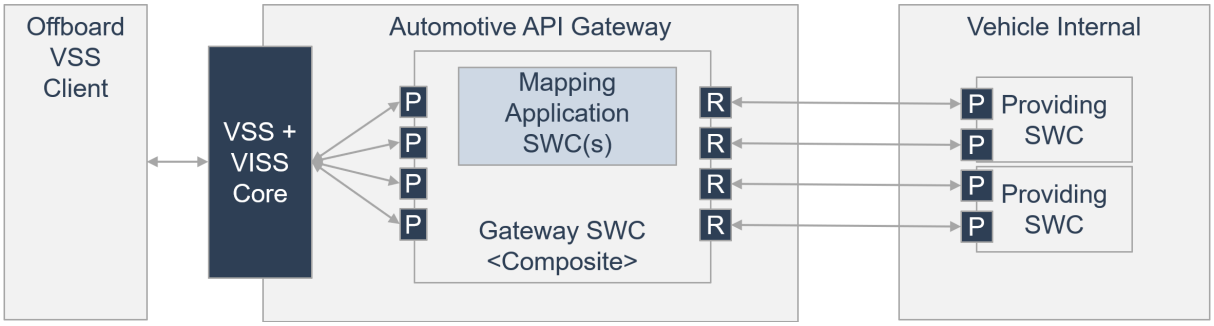


Figure 7.1: Conceptual Depiction Automotive API Gateway

The Automotive API Gateway provides an AUTOSAR compliant interface to provide VSS through VISS. Implementors must implement the provided skeletons. They may provide custom logic through Adaptive Applications to obtain VSS defined values to external entities, or they may utilize declarative approaches that utilize ARXML to describe the mapping of data entities.

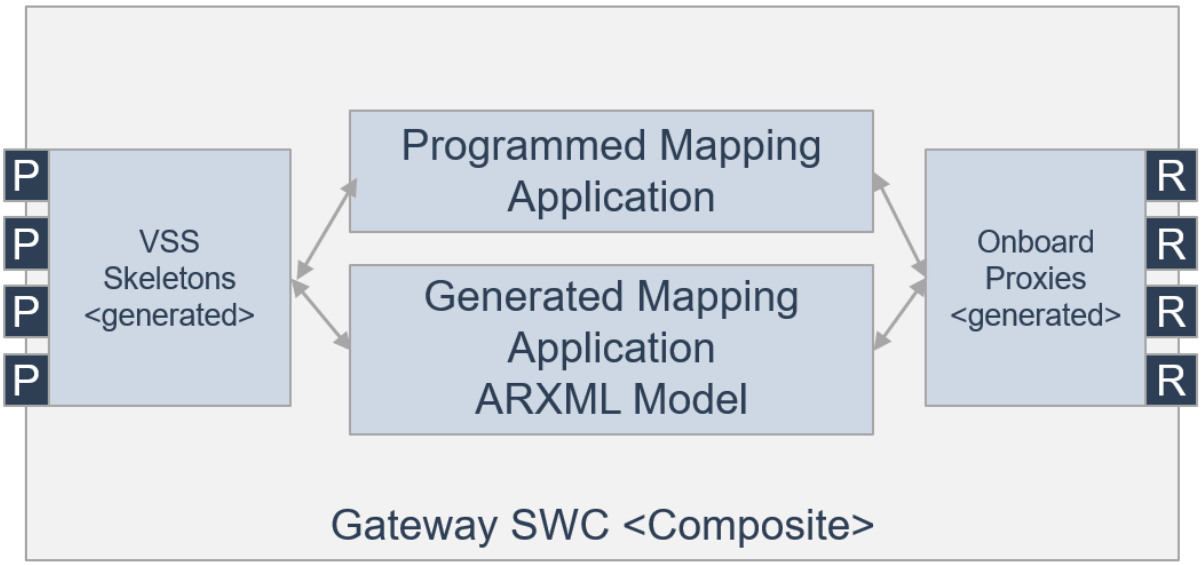


Figure 7.2: Mapping Alternatives

7.1 Gateway Software Component

The gateway software component models all VSS/VISS related provided ports per default. Providing the internal required ports are subject to an integration step. The VSS Representation does not generate required ports.

In case the gateway software component provides the whole catalog a single software component may suffice. However, there may be use cases where providing multiple gateways is reasonable for isolation purposes.

[AP_TR_VSS_05301] Gateway Software Component

Status: DRAFT

⌈The `VSS Importer` shall produce one or more `CompositionSwComponentType` for the gateway. Producing one or more `CompositionSwComponentType` is import specific and may be determined by implementation specific import configuration.⌋

[AP_TR_VSS_05302] Provided Ports Prototype

Status: DRAFT

⌈The `VSS Importer` shall produce a `PPortPrototype` for each `ProvidedUserDefinedServiceInstance`. Producing one or more `PPortPrototype` for one or more gateway software component is import specific and may be determined by implementation specific import configuration.⌋

[AP_TR_VSS_05303] Provided Ports Prototype Short Name

Status: DRAFT

⌈The `PPortPrototype` short name shall be the VSS instance path where '.' is to be replaced with underscores. If the VSS path is longer than 128 characters the last 40 characters shall be replaced with the 20 bytes of an SHA1 hash encoded as hexadecimal value of the entire VSS instance path after the replacement with underscores.⌋

[AP_TR_VSS_05306] Provided Ports Prototype Long Name

Status: DRAFT

⌈The `VSS Importer` shall produce a `LLongName`. The `VSS Importer` shall produce a `MultilanguageLongName` with the `LLongName` as its parent. The L attribute shall be set to FOR-ALL. The value shall be the VSS instance path where '.' is to be replaced with underscores.⌋

[AP_TR_VSS_05304] Provided Interface

Status: DRAFT

⌈The `VSS Importer` shall produce a PortInterfaceARRef reference to the respective VSS derived service interface type as providedInterface attribute of the `PPortPrototype`.⌋

[AP_TR_VSS_05305] Service Instance to Port Prototype Mapping

Status: DRAFT

⌈For each VSS derived service instance the `VSS Importer` shall produce a `ServiceInstanceToPortPrototypeMapping`. The short name of the element shall be set to <servicename>Mapping. The `VSS Importer` shall reference the corresponding portPrototype and the corresponding service instance.⌋

A Appendix

A.1 Mentioned Manifest Elements

This section contains the Manifest Elements mentioned in this documentation. It also 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	**ARPackage**			
Note	AUTOSAR package, allowing to create top level packages to structure the contained AUElements. ARPackages are open sets. This means that in a file based description system multiple files can be used to partially describe the contents of a package. This is an extended version of MSR's SW-SYSTEM.			
Base	*ARObject*, *AtpBlueprint*, *CollectableElement*, *Identifiable*, *MultilanguageReferrable*, *Referrable*			
Aggregated by	ARPackage.arPackage, AUTOSAR.arPackage			
Attribute	***Type***	***Mult.***	***Kind***	***Note***
arPackage	ARPackage	*	aggr	This represents a sub package within an ARPackage, thus allowing for an unlimited package hierarchy. **Stereotypes:** atpVariation **Tags:** atp.Splitkey=arPackage, arPackage.shortName, ar Package.variationPoint.shortLabel vh.latestBindingTime=blueprintDerivationTime xml.sequenceOffset=30
element	PackageableElement	*	aggr	Elements that are part of this package **Stereotypes:** atpVariation **Tags:** atp.Splitkey=element, element.shortName, element.variationPoint.shortLabel vh.latestBindingTime=systemDesignTime xml.sequenceOffset=20
referenceBase	ReferenceBase	*	aggr	This denotes the reference bases for the package. This is the basis for all relative references within the package. The base needs to be selected according to the base attribute within the references. **Tags:** atp.Splitkey=referenceBase, referenceBase.shortLabel xml.sequenceOffset=10

Table A.1: ARPackage

Class	**AdminData**
Note	AdminData represents the ability to express administrative information and custom extensions for an element. This administration information is to be treated as meta-data such as revision id or state of the file. There are basically the following kinds of meta-data • The language and/or used languages. • Revision information covering e.g. revision number, state, release date, changes. Note that this information can be given in general as well as related to a particular company. • Document meta-data specific for a company Beside that a custom extension of model-data is possible by • Special data

▽

△

Class	**AdminData**			
Base	*ARObject*			
Aggregated by	AUTOSAR.adminData, *Describable*.adminData, *Identifiable*.adminData			
Attribute	***Type***	***Mult.***	***Kind***	***Note***
docRevision (ordered)	DocRevision	*	aggr	This allows to denote information about the current revision of the object. Note that information about previous revisions can also be logged here. The entries shall be sorted descendant by date in order to reflect the history. Therefore the most recent entry representing the current version is denoted first. **Tags:** xml.roleElement=true xml.roleWrapperElement=true xml.sequenceOffset=50 xml.typeElement=false xml.typeWrapperElement=false
language	LEnum	0..1	attr	This attribute specifies the master language of the document or the document fragment. The master language is the one in which the document is maintained and from which the other languages are derived from. In particular in case of inconsistencies, the information in the master language is priority. **Tags:** xml.sequenceOffset=20
sdg	Sdg	*	aggr	This property allows to keep special data which is not represented by the standard model. It can be utilized to keep e.g. tool specific data. **Tags:** atp.Splitkey=sdg, sdg.sdgCaption.shortName xml.roleElement=true xml.roleWrapperElement=true xml.sequenceOffset=60 xml.typeElement=false xml.typeWrapperElement=false
usedLanguages	MultiLanguagePlainText	0..1	aggr	This property specifies the languages which are provided in the document. Therefore it should only be specified in the top level admin data. For each language provided in the document there is one entry in MultilanguagePlain Text. The content of each entry can be used for illustration of the language. The used language itself depends on the language attribute in the entry. **Tags:** atp.Splitkey=usedLanguages xml.sequenceOffset=30

Table A.2: AdminData

Class	**ApplicationArrayDataType**			
Note	An application data type which is an array, each element is of the same application data type. **Tags:** atp.recommendedPackage=ApplicationDataTypes			
Base	*ARElement*, *ARObject*, *ApplicationCompositeDataType*, *ApplicationDataType*, *AtpBlueprint*, *Atp Classifier*, *AtpType*, *AutosarDataType*, *CollectableElement*, *Identifiable*, *MultilanguageReferrable*, *PackageableElement*, *Referrable*			
Aggregated by	ARPackage.element			
Attribute	***Type***	***Mult.***	***Kind***	***Note***

▽

△

Class	**ApplicationArrayDataType**			
dynamicArray SizeProfile	String	0..1	attr	Specifies the profile which the array will follow if it is a variable size array.
element	ApplicationArray Element	0..1	aggr	This association implements the concept of an array element. That is, in some cases it is necessary to be able to identify single array elements, e.g. as input values for an interpolation routine. **Tags:** atp.Splitkey=element, element.shortName

Table A.3: ApplicationArrayDataType

Class	***ApplicationDataType*** (abstract)			
Note	`ApplicationDataType` defines a data type from the application point of view. Especially it should be used whenever something "physical" is at stake. An `ApplicationDataType` represents a set of values as seen in the application model, such as measurement units. It does not consider implementation details such as bit-size, endianess, etc. It should be possible to model the application level aspects of a VFB system by using `ApplicationDataType`s only.			
Base	*ARElement*, *ARObject*, *AtpBlueprint*, *AtpClassifier*, *AtpType*, *AutosarDataType*, *CollectableElement*, *Identifiable*, *MultilanguageReferrable*, *PackageableElement*, *Referrable*			
Subclasses	*ApplicationCompositeDataType*, ApplicationPrimitiveDataType			
Aggregated by	ARPackage.element			
Attribute	***Type***	***Mult.***	***Kind***	***Note***
–	–	–	–	–

Table A.4: ApplicationDataType

Class	**ApplicationPrimitiveDataType**			
Note	A primitive data type defines a set of allowed values. **Tags:** atp.recommendedPackage=ApplicationDataTypes			
Base	*ARElement*, *ARObject*, *ApplicationDataType*, *AtpBlueprint*, *AtpClassifier*, *AtpType*, *AutosarDataType*, *CollectableElement*, *Identifiable*, *MultilanguageReferrable*, *PackageableElement*, *Referrable*			
Aggregated by	ARPackage.element			
Attribute	***Type***	***Mult.***	***Kind***	***Note***
–	–	–	–	–

Table A.5: ApplicationPrimitiveDataType

Class	**ApplicationRecordDataType**			
Note	An application data type which can be decomposed into prototypes of other application data types. **Tags:** atp.recommendedPackage=ApplicationDataTypes			
Base	*ARElement*, *ARObject*, *ApplicationCompositeDataType*, *ApplicationDataType*, *AtpBlueprint*, *AtpClassifier*, *AtpType*, *AutosarDataType*, *CollectableElement*, *Identifiable*, *MultilanguageReferrable*, *PackageableElement*, *Referrable*			
Aggregated by	ARPackage.element			
Attribute	***Type***	***Mult.***	***Kind***	***Note***

▽

△

Class	**ApplicationRecordDataType**			
element (ordered)	ApplicationRecord Element	*	aggr	Specifies an element of a record. The aggregation of `ApplicationRecordElement` is subject to variability with the purpose to support the conditional existence of elements inside a `ApplicationRecordData` Type. **Stereotypes:** atpVariation **Tags:** vh.latestBindingTime=preCompileTime

Table A.6: ApplicationRecordDataType

Class	**ApplicationRecordElement**			
Note	Describes the properties of one particular element of an application record data type.			
Base	*ARObject*, *ApplicationCompositeElementDataPrototype*, *AtpFeature*, *AtpPrototype*, *DataPrototype*, *Identifiable*, *MultilanguageReferrable*, *Referrable*			
Aggregated by	ApplicationRecordDataType.element, *AtpClassifier*.atpFeature			
Attribute	***Type***	***Mult.***	***Kind***	***Note***
isOptional	Boolean	0..1	attr	This attribute represents the ability to declare the enclosing `ApplicationRecordElement` as optional. This means the that, at runtime, the `ApplicationRecordElement` may or may not have a valid value and shall therefore be ignored. The underlying runtime software provides means to set the `ApplicationRecordElement` as not valid at the sending end of a communication and determine its validity at the receiving end.

Table A.7: ApplicationRecordElement

Class	**ClientServerOperation**			
Note	An operation declared within the scope of a client/server interface.			
Base	*ARObject*, *AtpClassifier*, *AtpFeature*, *AtpStructureElement*, *Identifiable*, *MultilanguageReferrable*, *Referrable*			
Aggregated by	ApplicationInterface.command, *AtpClassifier*.atpFeature, ClientServerInterface.operation, DiagnosticDataElementInterface.read, DiagnosticDataIdentifierInterface.read, DiagnosticDataIdentifierInterface.write, DiagnosticExtendedDataRecordInterface.provide, DiagnosticRoutineInterface.requestResult, DiagnosticRoutineInterface.start, DiagnosticRoutineInterface.stop, ServiceInterface.method			
Attribute	***Type***	***Mult.***	***Kind***	***Note***
argument (ordered)	ArgumentDataPrototype	*	aggr	An argument of this `ClientServerOperation`. **Stereotypes:** atpVariation **Tags:** vh.latestBindingTime=blueprintDerivationTime
fireAndForget	Boolean	0..1	attr	This attribute defines whether this method is a fire&forget method (true) or not (false). This Attribute is only used by the AUTOSAR Adaptive Platform.
possibleApError	ApApplicationError	*	ref	This reference identifies AdaptivePlatformApplicationErrors as a possible error raised by the enclosing ClientServerOperation. **Tags:** atp.Splitkey=possibleApError This Attribute is only used by the AUTOSAR Adaptive Platform.

▽

△

Class	**ClientServerOperation**			
possibleApError Set	ApApplicationErrorSet	*	ref	This reference represents the ability to refer to an entire group of ApApplicationErrors as one model element instead of having to refer to all the represented Ap ApplicationErrors separately. **Tags:** atp.Splitkey=possibleApErrorSet This Attribute is only used by the AUTOSAR Adaptive Platform.

Table A.8: ClientServerOperation

Class	**CompositionSwComponentType**			
Note	A `CompositionSwComponentType` aggregates `SwComponentPrototype`s (that in turn are typed by `SwComponentType`)s as well as `SwConnector`s for primarily connecting `SwComponentPrototype`s among each others and towards the surface of the `CompositionSwComponentType`. By this means, a hierarchical structures of software-components can be created. **Tags:** atp.recommendedPackage=SwComponentTypes			
Base	*ARElement*, *ARObject*, *AtpBlueprint*, *AtpClassifier*, *AtpType*, *CollectableElement*, *Identifiable*, *MultilanguageReferrable*, *PackageableElement*, *Referrable*, *SwComponentType*			
Aggregated by	ARPackage.element			
Attribute	***Type***	***Mult.***	***Kind***	***Note***
component	SwComponent Prototype	*	aggr	The instantiated components that are part of this composition. **Stereotypes:** atpVariation **Tags:** atp.Splitkey=component, component.shortName, component.variationPoint.shortLabel vh.latestBindingTime=postBuild
connector	SwConnector	*	aggr	`SwConnector`s have the principal ability to establish a connection among `PortPrototype`s. They can have many roles in the context of a `CompositionSwComponentType`. Details are refined by subclasses. The aggregation of `SwConnector`s is subject to variability with the purpose to support variant data flow. **Stereotypes:** atpVariation **Tags:** atp.Splitkey=connector, connector.shortName, connector.variationPoint.shortLabel vh.latestBindingTime=postBuild
constantValue Mapping	ConstantSpecification MappingSet	*	ref	Reference to the `ConstantSpecificationMapping` to be applied for initValues of `PPortComSpec`s and `RPortComSpec`. **Tags:** atp.Splitkey=constantValueMapping
dataType Mapping	DataTypeMappingSet	*	ref	Reference to the `DataTypeMappingSet` to be applied for the used `ApplicationDataType`s in `ServiceInterface`s. **Tags:** atp.Splitkey=dataTypeMapping

▽

△

Class	CompositionSwComponentType			
physical Dimension Mapping	PhysicalDimension MappingSet	0..1	ref	This reference identifies the `PhysicalDimensionMappingSet` that is applicable in the context of the enclosing `CompositionSwComponentType`. The `PhysicalDimensionMapping`s contained in the `PhysicalDimensionMappingSet` shall be taken into account for the assessment of the compatibility of `PhysicalDimension`s in the context of creation of a `PortInterfaceMapping` in the scope of the `CompositionSwComponentType`. **Tags:** atp.Splitkey=physicalDimensionMapping

Table A.9: CompositionSwComponentType

Class	CompuConst			
Note	This meta-class represents the fact that the value of a computation method scale is constant.			
Base	*ARObject*			
Aggregated by	Compu.compuDefaultValue, CompuScale.compuInverseValue, CompuScaleConstantContents.compuConst			
Attribute	**Type**	**Mult.**	**Kind**	**Note**
compuConst ContentType	CompuConstContent	0..1	aggr	This is the actual content of the constant compu method scale. **Tags:** xml.roleElement=false xml.roleWrapperElement=false xml.sequenceOffset=10 xml.typeElement=false xml.typeWrapperElement=false

Table A.10: CompuConst

Class	CompuMethod			
Note	This meta-class represents the ability to express the relationship between a physical value and the mathematical representation. Note that this is still independent of the technical implementation in data types. It only specifies the formula how the internal value corresponds to its physical pendant. **Tags:** atp.recommendedPackage=CompuMethods			
Base	*ARElement*, *ARObject*, *AtpBlueprint*, *CollectableElement*, *Identifiable*, *MultilanguageReferrable*, *PackageableElement*, *Referrable*			
Aggregated by	ARPackage.element			
Attribute	**Type**	**Mult.**	**Kind**	**Note**
compuInternal ToPhys	Compu	0..1	aggr	This specifies the computation from internal values to physical values. **Tags:** atp.Splitkey=compuInternalToPhys xml.sequenceOffset=80
compuPhysTo Internal	Compu	0..1	aggr	This represents the computation from physical values to the internal values. **Tags:** atp.Splitkey=compuPhysToInternal xml.sequenceOffset=90

▽

△

Class	**CompuMethod**			
displayFormat	DisplayFormatString	0..1	attr	This property specifies, how the physical value shall be displayed e.g. in documents or measurement and calibration tools. **Tags:** xml.sequenceOffset=20
unit	Unit	0..1	ref	This is the physical unit of the Physical values for which the CompuMethod applies. **Tags:** atp.Splitkey=unit xml.sequenceOffset=30

Table A.11: CompuMethod

Class	**CompuScale**			
Note	This meta-class represents the ability to specify one segment of a segmented computation method.			
Base	*ARObject*			
Aggregated by	CompuScales.compuScale			
Attribute	***Type***	***Mult.***	***Kind***	***Note***
a2lDisplayText	String	0..1	attr	The value of this attribute shall be taken for generating one display text (specifically the OutVal) within the equivalent of the enclosing `CompuMethod` in A2L.
compuInverse Value	CompuConst	0..1	aggr	This is the inverse value of the constraint. This supports the case that the scale is not reversible per se. **Tags:** atp.Splitkey=compuInverseValue xml.sequenceOffset=60
compuScale Contents	CompuScaleContents	0..1	aggr	This represents the computation details of the scale. **Tags:** xml.roleElement=false xml.roleWrapperElement=false xml.sequenceOffset=70 xml.typeElement=false xml.typeWrapperElement=false
desc	MultiLanguageOverview Paragraph	0..1	aggr	<desc> represents a general but brief description of the object in question. **Tags:** atp.Splitkey=desc xml.sequenceOffset=30
lowerLimit	Limit	0..1	attr	This specifies the lower limit of the scale. **Stereotypes:** atpVariation **Tags:** vh.latestBindingTime=preCompileTime xml.sequenceOffset=40
mask	PositiveUnlimitedInteger	0..1	attr	In difference to all the other computational methods every COMPU-SCALE will be applied including the bit MASK. Therefore it is allowed for this type of COMPU-METHOD, that COMPU-SCALES overlap. To calculate the string reverse to a value, the string has to be split and the according value for each substring has to be summed up. The sum is finally transmitted. The processing has to be done in order of the COMPU-SCALE elements. **Tags:** xml.sequenceOffset=35

▽

△

Class	**CompuScale**			
shortLabel	Identifier	0..1	attr	This element specifies a short name for the particular scale. The name can for example be used to derive a programming language identifier. **Tags:** xml.sequenceOffset=20
symbol	CIdentifier	0..1	attr	The symbol, if provided, is used by code generators to get a C identifier for the CompuScale. The name will be used as is for the code generation, therefore it needs to be unique within the generation context. **Tags:** xml.sequenceOffset=25
upperLimit	Limit	0..1	attr	This specifies the upper limit of a of the scale. **Stereotypes:** atpVariation **Tags:** vh.latestBindingTime=preCompileTime xml.sequenceOffset=50

Table A.12: CompuScale

Class	**CppTemplateArgument**			
Note	This meta-class has the ability to define properties for template arguments. This Class is only used by the AUTOSAR Adaptive Platform.			
Base	*ARObject*			
Aggregated by	*CppImplementationDataType*.templateArgument			
Attribute	*Type*	*Mult.*	*Kind*	*Note*
allocator	Allocator	0..1	ref	This reference identifies the applicable allocator.
category	CategoryString	0..1	attr	This attribute shall be used to contribute further clarification regarding the semantics of the enclosing Cpp TemplateArgument.
inplace	Boolean	0..1	attr	This attribute specifies whether the shortName of the referenced templateType is used in the code generation and the type declaration is defined outside of the enclosing CppImplementationDataType (true) or whether the type definition is embedded inside of the enclosing CppImplementationDataType and the shortName is ignored (false).
templateType	CppImplementation DataType	0..1	ref	This reference identifies the data type of the specific template argument required for the language binding.

Table A.13: CppTemplateArgument

Class	**DataTypeMap**			
Note	This class represents the relationship between `ApplicationDataType` and its implementing `AbstractImplementationDataType`.			
Base	*ARObject*			
Aggregated by	DataTypeMappingSet.dataTypeMap			
Attribute	*Type*	*Mult.*	*Kind*	*Note*
applicationData Type	ApplicationDataType	0..1	ref	This is the corresponding `ApplicationDataType` **Tags:** atp.Splitkey=applicationDataType

▽

△

Class	**DataTypeMap**			
implementation DataType	AbstractImplementation DataType	0..1	ref	This is the corresponding `AbstractImplementationDataType`. **Tags:** atp.Splitkey=implementationDataType

Table A.14: DataTypeMap

Class	**DataTypeMappingSet**			
Note	This class represents a list of mappings between `ApplicationDataType`s and `ImplementationDataType`s. In addition, it can contain mappings between `ImplementationDataType`s and `ModeDeclarationGroup`s. **Tags:** atp.recommendedPackage=DataTypeMappingSets			
Base	*ARElement*, *ARObject*, *AtpBlueprint*, *CollectableElement*, *Identifiable*, *MultilanguageReferrable*, *PackageableElement*, *Referrable*			
Aggregated by	ARPackage.element			
Attribute	***Type***	***Mult.***	***Kind***	***Note***
dataTypeMap	DataTypeMap	*	aggr	This is one particular association between an `ApplicationDataType` and its `AbstractImplementationDataType`. **Tags:** atp.Splitkey=dataTypeMap
modeRequest TypeMap	ModeRequestTypeMap	*	aggr	This is one particular association between an `ModeDeclarationGroup` and its `AbstractImplementationDataType`. **Tags:** atp.Splitkey=modeRequestTypeMap

Table A.15: DataTypeMappingSet

Class	«atpMixed» **DocumentationBlock**			
Note	This class represents a documentation block. It is made of basic text structure elements which can be displayed in a table cell.			
Base	*ARObject*			
Aggregated by	ApplicabilityInfo.remark, AUTOSAR.introduction, BlueprintGenerator.introduction, *BlueprintPolicy* *Modifiable*.blueprintDerivationGuide, *DataMapping*.introduction, DefItem.def, *Describable*.introduction, EcucAddInfoParamValue.value, EcuResourceEstimation.introduction, Entry.entryContents, Frame Mapping.introduction, *GeneralAnnotation*.annotationText, *Identifiable*.introduction, IPdu Mapping.introduction, ISignalMapping.introduction, Item.itemContents, LabeledItem.itemContents, Life CycleInfo.remark, *MappingConstraint*.introduction, MsrQueryP2.msrQueryResultP2, Note.noteText, *ScheduleTableEntry*.introduction, *SignalPathConstraint*.introduction, StructuredReq.conflicts, Structured Req.dependencies, StructuredReq.description, StructuredReq.rationale, StructuredReq.remark, StructuredReq.supportingMaterial, StructuredReq.useCase, SwAxisType.swGenericAxisDesc, Topic Content.blockLevelContent, TraceableText.text, VariationPoint.blueprintCondition			
Attribute	***Type***	***Mult.***	***Kind***	***Note***
defList	DefList	0..1	aggr	This represents a definition list in the documentation block. **Stereotypes:** atpVariation **Tags:** vh.latestBindingTime=postBuild xml.sequenceOffset=40
figure	MIFigure	0..1	aggr	This represents a figure in the documentation block. **Stereotypes:** atpVariation **Tags:** vh.latestBindingTime=postBuild xml.sequenceOffset=70

▽

△

Class	«atpMixed» **DocumentationBlock**			
formula	MIFormula	0..1	aggr	This is a formula in the definition block. **Stereotypes:** atpVariation **Tags:** vh.latestBindingTime=postBuild xml.sequenceOffset=60
labeledList	LabeledList	0..1	aggr	This represents a labeled list. **Stereotypes:** atpVariation **Tags:** vh.latestBindingTime=postBuild xml.sequenceOffset=50
list	List	0..1	aggr	This represents numbered or unnumbered list. **Stereotypes:** atpVariation **Tags:** vh.latestBindingTime=postBuild xml.sequenceOffset=30
msrQueryP2	MsrQueryP2	0..1	aggr	This represents automatically contributed contents provided by an msrquery in the context of Documentation Block.
note	Note	0..1	aggr	This represents a note in the text flow. **Stereotypes:** atpVariation **Tags:** vh.latestBindingTime=postBuild xml.sequenceOffset=80
p	MultiLanguage Paragraph	0..1	aggr	This is one particular paragraph. **Stereotypes:** atpVariation **Tags:** vh.latestBindingTime=postBuild xml.sequenceOffset=10
structuredReq	StructuredReq	0..1	aggr	This aggregation supports structured requirements embedded in a documentation block. **Stereotypes:** atpVariation **Tags:** vh.latestBindingTime=postBuild xml.sequenceOffset=100
trace	TraceableText	0..1	aggr	This represents traceable text in the documentation block. This allows to specify requirements/constraints in any documentation block. The kind of the trace is specified in the category. **Stereotypes:** atpVariation **Tags:** vh.latestBindingTime=postBuild xml.sequenceOffset=90
verbatim	MultiLanguageVerbatim	0..1	aggr	This represents one particular verbatim text. **Stereotypes:** atpVariation **Tags:** vh.latestBindingTime=postBuild xml.sequenceOffset=20

Table A.16: DocumentationBlock

Class	**Field**
Note	This meta-class represents the ability to define a piece of data that can be accessed with read and/or write semantics. It is also possible to generate a notification if the value of the data changes. This Class is only used by the AUTOSAR Adaptive Platform.

▽

△

Class	**Field**			
Base	*ARObject*, *AtpFeature*, *AtpPrototype*, *AutosarDataPrototype*, *DataPrototype*, *Identifiable*, *Multilanguage Referrable*, *Referrable*			
Aggregated by	ApplicationInterface.attribute, *AtpClassifier*.atpFeature, ServiceInterface.field			
Attribute	***Type***	***Mult.***	***Kind***	***Note***
hasGetter	Boolean	0..1	attr	This attribute controls whether read access is foreseen to this field.
hasNotifier	Boolean	0..1	attr	This attribute controls whether a notification semantics is foreseen to this field.
hasSetter	Boolean	0..1	attr	This attribute controls whether write access is foreseen to this field.

Table A.17: Field

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 ComponentPrototype, RootSwCompositionPrototype, RptComponent, RptContainer, RptExecutable Entity, RptExecutableEntityEvent, RptExecutionContext, RptProfile, RptServicePoint, RunnableEntity Group, *SdgAttribute*, SdgClass, SecOcJobMapping, SecOcJobRequirement, SecureCommunication AuthenticationProps, *SecureCommunicationDeployment*, SecureCommunicationFreshnessProps, SecurityEventContextDataElement, SecurityEventContextProps, ServerComSpecProps, *Service* ▽

▽

△

Class	***Identifiable*** (abstract)			
	△ *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 A.18: Identifiable

Primitive	**Identifier**			
Note	An Identifier is a string with a number of constraints on its appearance, satisfying the requirements typical programming languages define for their Identifiers. This datatype represents a string, that can be used as a c-Identifier. It shall start with a letter, may consist of letters, digits and underscores. **Tags:** xml.xsd.customType=IDENTIFIER xml.xsd.maxLength=128 xml.xsd.pattern=[a-zA-Z][a-zA-Z0-9_]* xml.xsd.type=string			
Attribute	***Type***	***Mult.***	***Kind***	***Note***
blueprintValue	String	0..1	attr	This represents a description that documents how the value shall be defined when deriving objects from the blueprint. **Tags:** atp.Status=draft xml.attribute=true
namePattern	String	0..1	attr	This attribute represents a pattern which shall be used to define the value of the identifier if the identifier in question is part of a blueprint. For more details refer to TPS_StandardizationTemplate. **Tags:** xml.attribute=true

Table A.19: Identifier

Class	«atpMixedString» **LLongName**			
Note	MixedContentForLongNames in one particular language. The language is denoted in the attribute l.			
Base	*ARObject*, *LanguageSpecific*, *MixedContentForLongName*			
Aggregated by	MultilanguageLongName.l4			
Attribute	***Type***	***Mult.***	***Kind***	***Note***

▽

△

Class	«atpMixedString» **LLongName**			
blueprintValue	String	0..1	attr	This represents a description that documents how the value shall be defined when deriving objects from the blueprint. **Tags:** atp.Status=draft xml.attribute=true

Table A.20: LLongName

Class	«atpMixedString» **LOverviewParagraph**			
Note	MixedContentForOverviewParagraph in one particular language. The language is denoted in the attribute l.			
Base	*ARObject*, *LanguageSpecific*, *MixedContentForOverviewParagraph*			
Aggregated by	IndentSample.l2, MultiLanguageOverviewParagraph.l2			
Attribute	***Type***	***Mult.***	***Kind***	***Note***
blueprintValue	String	0..1	attr	This represents a description that documents how the value shall be defined when deriving objects from the blueprint. **Tags:** atp.Status=draft xml.attribute=true

Table A.21: LOverviewParagraph

Class	«atpMixedString» ***MixedContentForOverviewParagraph*** (abstract)			
Note	This is the text model of a restricted paragraph item within a documentation. Such restricted paragraphs are used mainly for overview items, e.g. desc.			
Base	*ARObject*			
Subclasses	LOverviewParagraph, SlOverviewParagraph			
Attribute	***Type***	***Mult.***	***Kind***	***Note***
br	Br	0..1	aggr	This element is the same as function here as in a HTML document i.e. it forces a line break.
e	EmphasisText	0..1	aggr	This is emphasis text. **Tags:** xml.sequenceOffset=60
ft	SlOverviewParagraph	0..1	aggr	This is a foot note within a paragraph.
ie	IndexEntry	0..1	aggr	This is an index entry. **Tags:** xml.sequenceOffset=100
sub	Superscript	0..1	attr	This is superscript text. **Tags:** xml.sequenceOffset=90
sup	Superscript	0..1	attr	This is subscript text. **Tags:** xml.sequenceOffset=80
trace	Traceable	0..1	ref	This allows to place an arbitrary reference to a traceable object in documentation.
tt	Tt	0..1	aggr	This is a technical term. **Tags:** xml.sequenceOffset=30
xref	Xref	0..1	aggr	This is a cross reference. **Tags:** xml.sequenceOffset=40

▽

△

Class	«atpMixedString» ***MixedContentForOverviewParagraph*** (abstract)			
xrefTarget	XrefTarget	0..1	aggr	This element specifies a reference target which can be scattered throughout the text. **Tags:** xml.sequenceOffset=50

Table A.22: MixedContentForOverviewParagraph

Class	**MultiLanguageOverviewParagraph**			
Note	This is the content of a multilingual paragraph in an overview item.			
Base	*ARObject*			
Aggregated by	Caption.desc, CompuScale.desc, *Describable*.desc, *Identifiable*.desc, LabeledItem.itemLabel, Modification.change, Modification.reason, SdgCaption.desc, SwRecordLayoutGroup.desc, SwRecord LayoutV.desc, VariationPoint.desc			
Attribute	***Type***	***Mult.***	***Kind***	***Note***
l2	LOverviewParagraph	1..*	aggr	This represents the text in one particular language. **Tags:** atp.Splitkey=l2 xml.roleElement=true xml.roleWrapperElement=false xml.sequenceOffset=20 xml.typeElement=false xml.typeWrapperElement=false

Table A.23: MultiLanguageOverviewParagraph

Class	**MultilanguageLongName**			
Note	This meta-class represents the ability to specify a long name which acts in the role of a headline. It is intended for human readers. Per language it should be around max 80 characters.			
Base	*ARObject*			
Aggregated by	AliasNameAssignment.label, *GeneralAnnotation*.label, *MultilanguageReferrable*.longName, Note.label, ValueGroup.label			
Attribute	***Type***	***Mult.***	***Kind***	***Note***
l4	LLongName	*	aggr	This is the long name in one particular language. **Tags:** atp.Splitkey=l4 xml.roleElement=true xml.roleWrapperElement=false xml.sequenceOffset=20 xml.typeElement=false xml.typeWrapperElement=false

Table A.24: MultilanguageLongName

Class	**Note**			
Note	This represents a note in a documentation, which may be used to highlight specific issues such as hints or caution notes. N.B., Documentation notes can be nested recursively, even if this is not really intended. In case of nested notes e.g. the note icon of inner notes might be omitted while rendering the note.			
Base	*ARObject*, *DocumentViewSelectable*, *Paginateable*			
Aggregated by	DocumentationBlock.note			
Attribute	***Type***	***Mult.***	***Kind***	***Note***

▽

△

Class	Note			
label	MultilanguageLongName	0..1	aggr	This label can be used to superseed the default label specified by the noteType attribute. It is in particular useful for noteType="other". **Tags:** xml.sequenceOffset=20
noteText	DocumentationBlock	0..1	aggr	This is the text content of the note. **Tags:** xml.roleElement=false xml.roleWrapperElement=false xml.sequenceOffset=30 xml.typeElement=false xml.typeWrapperElement=false
noteType	NoteTypeEnum	0..1	attr	Type of the Note. Default is "HINT" **Tags:** xml.attribute=true

Table A.25: Note

Class	PPortPrototype			
Note	Component port providing a certain port interface.			
Base	*ARObject*, *AbstractProvidedPortPrototype*, *AtpFeature*, *AtpPrototype*, *Identifiable*, *MultilanguageReferrable*, *PortPrototype*, *Referrable*			
Aggregated by	*AtpClassifier*.atpFeature, *SwComponentType*.port			
Attribute	**Type**	**Mult.**	**Kind**	**Note**
provided Interface	PortInterface	0..1	tref	The interface that this port provides. **Stereotypes:** isOfType **Tags:** atp.Splitkey=providedInterface

Table A.26: PPortPrototype

Class	PhysConstrs			
Note	This meta-class represents the ability to express physical constraints. Therefore it has (in opposite to InternalConstrs) a reference to a Unit.			
Base	*ARObject*			
Aggregated by	DataConstrRule.physConstrs			
Attribute	**Type**	**Mult.**	**Kind**	**Note**
lowerLimit	Limit	0..1	attr	This specifies the lower limit of the constraint. **Stereotypes:** atpVariation **Tags:** vh.latestBindingTime=preCompileTime xml.sequenceOffset=20
maxDiff	Numerical	0..1	attr	Maximum difference that is permitted between two consecutive values if the constraint is applied to an axis. **Tags:** xml.sequenceOffset=60
maxGradient	Numerical	0..1	attr	This element specifies the maximum slope that may be used in curves and maps. **Tags:** xml.sequenceOffset=50
monotony	MonotonyEnum	0..1	attr	This specifies the monotony constraints on the data object. Note that this applies only to curves and maps. **Tags:** xml.sequenceOffset=70

▽

△

Class	**PhysConstrs**			
unit	Unit	0..1	ref	This is the unit to which the physical constraints relate to. In particular, it is the physical unit of the specified limits. **Tags:** atp.Splitkey=unit xml.sequenceOffset=80
upperLimit	Limit	0..1	attr	This specifies the upper limit of the constraint. **Stereotypes:** atpVariation **Tags:** vh.latestBindingTime=preCompileTime xml.sequenceOffset=30

Table A.27: PhysConstrs

Class	«atpMixedString» **PositiveIntegerValueVariationPoint**			
Note	This class represents an attribute value variation point for positive Integer attributes. Note that this class might be used in the extended meta-model only.			
Base	*ARObject*, *AttributeValueVariationPoint*, *FormulaExpression*, *SwSystemconstDependentFormula*			
Aggregated by	VariationPointProxy.valueAccess			
Attribute	*Type*	*Mult.*	*Kind*	*Note*
–	–	–	–	–

Table A.28: PositiveIntegerValueVariationPoint

Class	**ProvidedUserDefinedServiceInstance**			
Note	This meta-class represents the ability to describe the existence and configuration of a provided service instance in a concrete implementation that is not standardized by AUTOSAR. **Tags:** atp.recommendedPackage=ServiceInstances This Class is only used by the AUTOSAR Adaptive Platform.			
Base	*ARElement*, *ARObject*, *AdaptivePlatformServiceInstance*, *CollectableElement*, *Identifiable*, *MultilanguageReferrable*, *PackageableElement*, *ProvidedApServiceInstance*, *Referrable*, *Uploadable DesignElement*, *UploadablePackageElement*			
Aggregated by	ARPackage.element			
Attribute	*Type*	*Mult.*	*Kind*	*Note*
–	–	–	–	–

Table A.29: ProvidedUserDefinedServiceInstance

Class	**Sd**			
Note	This class represents a primitive element in a special data group.			
Base	*ARObject*			
Aggregated by	SdgContents.sd			
Attribute	*Type*	*Mult.*	*Kind*	*Note*
gid	NameToken	0..1	attr	This attributes specifies an identifier. Gid comes from the SGML/XML-Term "Generic Identifier" which is the element name in XML. The role of this attribute is the same as the name of an XML - element. **Tags:** xml.attribute=true

▽

△

Class	**Sd**			
value	VerbatimStringPlain	0..1	attr	This is the value of the special data. **Tags:** xml.roleElement=false xml.roleWrapperElement=false xml.typeElement=false xml.typeWrapperElement=false
xmlSpace	XmlSpaceEnum	0..1	attr	This attribute is used to signal an intention that in that element, white space should be preserved by applications. It is defined according to xml:space as declared by W3C. **Tags:** xml.attribute=true xml.attributeRef=true xml.enforceMinMultiplicity=true xml.name=space xml.nsPrefix=xml

Table A.30: Sd

Class	**Sdg**			
Note	Sdg (SpecialDataGroup) is a generic model which can be used to keep arbitrary information which is not explicitly modeled in the meta-model. Sdg can have various contents as defined by sdgContentsType. Special Data should only be used moderately since all elements should be defined in the meta-model. Thereby SDG should be considered as a temporary solution when no explicit model is available. If an sdg Caption is available, it is possible to establish a reference to the sdg structure.			
Base	*ARObject*			
Aggregated by	AdminData.sdg, BuildActionEnvironment.sdg, BuildActionInvocator.sdg, BuildActionIoElement.sdg, File InfoComment.sdg, RptHook.sdg, SdgContents.sdg, VariationPoint.sdg			
Attribute	***Type***	***Mult.***	***Kind***	***Note***
gid	NameToken	0..1	attr	This attributes specifies an identifier. Gid comes from the SGML/XML-Term "Generic Identifier" which is the element name in XML. The role of this attribute is the same as the name of an XML - element. **Tags:** xml.attribute=true
sdgCaption	SdgCaption	0..1	aggr	This aggregation allows to assign the properties of Identifiable to the sdg. By this, a shortName etc. can be assigned to the Sdg. **Stereotypes:** atpIdentityContributor **Tags:** xml.sequenceOffset=20
sdgContents Type	SdgContents	0..1	aggr	This is the content of the Sdg. **Tags:** xml.roleElement=false xml.roleWrapperElement=false xml.sequenceOffset=30 xml.typeElement=false xml.typeWrapperElement=false

Table A.31: Sdg

Class	**ServiceInstanceToPortPrototypeMapping**			
Note	This meta-class represents the ability to assign a transport layer dependent ServiceInstance to a Port Prototype. With this mapping it is possible to define how specific PortPrototypes are represented in the middleware in terms of service configuration. **Tags:** atp.recommendedPackage=ServiceInstanceToPortPrototypeMappings This Class is only used by the AUTOSAR Adaptive Platform.			
Base	*ARElement*, *ARObject*, *CollectableElement*, *Identifiable*, *MultilanguageReferrable*, *Packageable Element*, *Referrable*, *UploadableDesignElement*, *UploadablePackageElement*			
Aggregated by	ARPackage.element			
Attribute	***Type***	***Mult.***	***Kind***	***Note***
clientId	PositiveInteger	0..1	attr	This attribute defines the client identifier for a user of a Service Interface (i.e. RPortPrototype or RequriredAp ServiceInstance)
portPrototype	PortPrototype	0..1	iref	Reference to a specific PortPrototype that represents the ServiceInstance. **Stereotypes:** atpUriDef **Tags:** atp.Splitkey=portPrototype, portPrototype.context RootSwComponentPrototype, portPrototype.context ComponentPrototype, portPrototype.targetPortPrototype **InstanceRef implemented by:** PortPrototypeIn ExecutableInstanceRef
process	Process	0..1	ref	Reference to the Process in which the enclosing Service InstanceToPortPrototypeMapping is executed. **Tags:** atp.Splitkey=process
processDesign	ProcessDesign	0..1	ref	Reference to the ProcessDesign in which the Executable that contains the SoftwareComponent and the referenced PortPrototype is executed. **Stereotypes:** atpUriDef **Tags:** atp.Splitkey=processDesign
serviceInstance	AdaptivePlatform ServiceInstance	0..1	ref	Reference to a ServiceInstance that is represented in the Software Component by the mapped group of Port Prototypes. **Tags:** atp.Splitkey=serviceInstance

Table A.32: ServiceInstanceToPortPrototypeMapping

Class	**ServiceInterface**			
Note	This represents the ability to define a PortInterface that consists of a heterogeneous collection of methods, events and fields. **Tags:** atp.recommendedPackage=ServiceInterfaces This Class is only used by the AUTOSAR Adaptive Platform.			
Base	*ARElement*, *ARObject*, *AtpBlueprint*, *AtpClassifier*, *AtpType*, *CollectableElement*, *Identifiable*, *MultilanguageReferrable*, *PackageableElement*, *PortInterface*, *Referrable*			
Aggregated by	ARPackage.element			
Attribute	***Type***	***Mult.***	***Kind***	***Note***
event	VariableDataPrototype	*	aggr	This represents the collection of events defined in the context of a ServiceInterface. **Stereotypes:** atpVariation **Tags:** atp.Splitkey=event, event.shortName, event.variation Point.shortLabel vh.latestBindingTime=blueprintDerivationTime xml.sequenceOffset=30



△

Class	**ServiceInterface**			
field	Field	*	aggr	This represents the collection of fields defined in the context of a ServiceInterface. **Stereotypes:** atpVariation **Tags:** atp.Splitkey=field, field.shortName, field.variation Point.shortLabel vh.latestBindingTime=blueprintDerivationTime xml.sequenceOffset=40
majorVersion	PositiveInteger	0..1	attr	Major version of the service contract. **Tags:** xml.sequenceOffset=10
method	ClientServerOperation	*	aggr	This represents the collection of methods defined in the context of a ServiceInterface. **Stereotypes:** atpVariation **Tags:** atp.Splitkey=method, method.shortName, method.variationPoint.shortLabel vh.latestBindingTime=blueprintDerivationTime xml.sequenceOffset=50
minorVersion	PositiveInteger	0..1	attr	Minor version of the service contract. **Tags:** xml.sequenceOffset=20
trigger	Trigger	*	aggr	This represents the collection of triggers defined in the context of a ServiceInterface. **Stereotypes:** atpVariation **Tags:** atp.Splitkey=trigger, trigger.shortName, trigger.variation Point.shortLabel vh.latestBindingTime=blueprintDerivationTime xml.sequenceOffset=60

Table A.33: ServiceInterface

Class	**StdCppImplementationDataType**			
Note	This meta-class represents the way to specify a data type definition that is taken as the basis for a C++ language binding to a C++ Standard Library feature. **Tags:** atp.recommendedPackage=CppImplementationDataTypes This Class is only used by the AUTOSAR Adaptive Platform.			
Base	*ARElement*, *ARObject*, *AbstractImplementationDataType*, *AtpBlueprint*, *AtpClassifier*, *AtpType*, *Autosar DataType*, *CollectableElement*, *CppImplementationDataType*, *CppImplementationDataTypeContext Target*, *Identifiable*, *MultilanguageReferrable*, *PackageableElement*, *Referrable*			
Aggregated by	ARPackage.element			
Attribute	***Type***	***Mult.***	***Kind***	***Note***
–	–	–	–	–

Table A.34: StdCppImplementationDataType

Class	«atpVariation» **SwDataDefProps**
Note	This class is a collection of properties relevant for data objects under various aspects. One could consider this class as a "pattern of inheritance by aggregation". The properties can be applied to all objects of all classes in which SwDataDefProps is aggregated. Note that not all of the attributes or associated elements are useful all of the time. Hence, the process definition (e.g. expressed with an OCL or a Document Control Instance MSR-DCI) has the task of implementing limitations. SwDataDefProps covers various aspects: • Structure of the data element for calibration use cases: is it a single value, a curve, or a map, but also the recordLayouts which specify how such elements are mapped/converted to the DataTypes in the programming language (or in AUTOSAR). This is mainly expressed by properties like swRecordLayout and swCalprmAxisSet • Implementation aspects, mainly expressed by swImplPolicy, swAddrMethod, baseType, implementationDataType and additionalNativeTypeQualifier • Access policy for the MCD system, mainly expressed by swCalibrationAccess • Semantics of the data element, mainly expressed by compuMethod and/or unit, dataConstr, invalid Value • Code generation policy provided by swRecordLayout **Tags:** vh.latestBindingTime=codeGenerationTime
Base	*ARObject*
Aggregated by	*AutosarDataType*.swDataDefProps, CompositeNetworkRepresentation.networkRepresentation, Cpp ImplementationDataTypeElement.swDataDefProps, *DataPrototype*.swDataDefProps, DataPrototype TransformationProps.networkRepresentationProps, DiagnosticDataElement.swDataDefProps, Diagnostic EnvDataElementCondition.swDataDefProps, DiagnosticExtendedDataRecordElement.swDataDefProps, DiagnosticSovdPrimitiveContentElement.swDataDefProps, DltArgumentProps.networkRep, FlatInstance Descriptor.swDataDefProps, ImplementationDataTypeElement.swDataDefProps, InstantiationDataDef Props.swDataDefProps, ISignal.networkRepresentationProps, McDataInstance.resultingProperties, ParameterAccess.swDataDefProps, PerInstanceMemory.swDataDefProps, *ReceiverComSpec*.network Representation, SecurityEventContextDataElement.networkRepresentation, *SenderComSpec*.network Representation, SomeipDataPrototypeTransformationProps.networkRepresentation, SwPointerTarget Props.swDataDefProps, SwServiceArg.swDataDefProps, SwSystemconst.swDataDefProps, System Signal.physicalProps

Attribute	**Type**	**Mult.**	**Kind**	**Note**
additionalNative TypeQualifier	NativeDeclarationString	0..1	attr	This attribute is used to declare native qualifiers of the programming language which can neither be deduced from the baseType (e.g. because the data object describes a pointer) nor from other more abstract attributes. Examples are qualifiers like "volatile", "strict" or "enum" of the C-language. All such declarations have to be put into one string. **Tags:** xml.sequenceOffset=235
annotation	Annotation	*	aggr	This aggregation allows to add annotations (yellow pads ...) related to the current data object. **Tags:** atp.Splitkey=annotation xml.roleElement=true xml.roleWrapperElement=true xml.sequenceOffset=20 xml.typeElement=false xml.typeWrapperElement=false
baseType	SwBaseType	0..1	ref	Base type associated with the containing data object. **Tags:** atp.Splitkey=baseType xml.sequenceOffset=50

▽

△

Class	«atpVariation» **SwDataDefProps**			
compuMethod	CompuMethod	0..1	ref	Computation method associated with the semantics of this data object. **Tags:** atp.Splitkey=compuMethod xml.sequenceOffset=180
dataConstr	DataConstr	0..1	ref	Data constraint for this data object. **Tags:** atp.Splitkey=dataConstr xml.sequenceOffset=190
displayFormat	DisplayFormatString	0..1	attr	This property describes how a number is to be rendered e.g. in documents or in a measurement and calibration system. **Tags:** xml.sequenceOffset=210
display Presentation	DisplayPresentation Enum	0..1	attr	This attribute controls the presentation of the related data for measurement and calibration tools.
implementation DataType	AbstractImplementation DataType	0..1	ref	This association denotes the ImplementationDataType of a data declaration via its aggregated SwDataDefProps. It is used whenever a data declaration is not directly referring to a base type. Especially • redefinition of an ImplementationDataType via a "typedef" to another ImplementationDatatype • the target type of a pointer (see SwPointerTarget Props), if it does not refer to a base type directly • the data type of an array or record element within an ImplementationDataType, if it does not refer to a base type directly • the data type of an SwServiceArg, if it does not refer to a base type directly **Tags:** atp.Splitkey=implementationDataType xml.sequenceOffset=215
invalidValue	ValueSpecification	0..1	aggr	Optional value to express invalidity of the actual data element. **Tags:** atp.Splitkey=invalidValue xml.sequenceOffset=255
stepSize	Float	0..1	attr	This attribute can be used to define a value which is added to or subtracted from the value of a DataPrototype when using up/down keys while calibrating.
swAddrMethod	SwAddrMethod	0..1	ref	Addressing method related to this data object. Via an association to the same SwAddrMethod it can be specified that several DataPrototypes shall be located in the same memory without already specifying the memory section itself. **Tags:** atp.Splitkey=swAddrMethod xml.sequenceOffset=30
swAlignment	AlignmentType	0..1	attr	The attribute describes the intended typical alignment of the DataPrototype. If the attribute is not defined the alignment is determined by the swBaseType size and the memoryAllocationKeywordPolicy of the referenced Sw AddrMethod. **Tags:** xml.sequenceOffset=33

▽

△

Class	«atpVariation» **SwDataDefProps**			
swBit Representation	SwBitRepresentation	0..1	aggr	Description of the binary representation in case of a bit variable. **Tags:** atp.Splitkey=swBitRepresentation xml.sequenceOffset=60
swCalibration Access	SwCalibrationAccess Enum	0..1	attr	Specifies the read or write access by MCD tools for this data object. **Tags:** xml.sequenceOffset=70
swCalprmAxis Set	SwCalprmAxisSet	0..1	aggr	This specifies the properties of the axes in case of a curve or map etc. This is mainly applicable to calibration parameters. **Tags:** atp.Splitkey=swCalprmAxisSet xml.sequenceOffset=90
swComparison Variable	SwVariableRefProxy	*	aggr	Variables used for comparison in an MCD process. **Tags:** atp.Splitkey=swComparisonVariable xml.sequenceOffset=170 xml.typeElement=false
swData Dependency	SwDataDependency	0..1	aggr	Describes how the value of the data object has to be calculated from the value of another data object (by the MCD system). **Tags:** atp.Splitkey=swDataDependency xml.sequenceOffset=200
swHostVariable	SwVariableRefProxy	0..1	aggr	Contains a reference to a variable which serves as a host-variable for a bit variable. Only applicable to bit objects. **Tags:** atp.Splitkey=swHostVariable xml.sequenceOffset=220 xml.typeElement=false
swImplPolicy	SwImplPolicyEnum	0..1	attr	Implementation policy for this data object. **Tags:** xml.sequenceOffset=230
swIntended Resolution	Numerical	0..1	attr	The purpose of this element is to describe the requested quantization of data objects early on in the design process. The resolution ultimately occurs via the conversion formula present (compuMethod), which specifies the transition from the physical world to the standardized world (and vice-versa) (here, "the slope per bit" is present implicitly in the conversion formula). In the case of a development phase without a fixed conversion formula, a pre-specification can occur through swIntendedResolution. The resolution is specified in the physical domain according to the property "unit". **Tags:** xml.sequenceOffset=240
swInterpolation Method	Identifier	0..1	attr	This is a keyword identifying the mathematical method to be applied for interpolation. The keyword needs to be related to the interpolation routine which needs to be invoked. **Tags:** xml.sequenceOffset=250

▽

△

Class	«atpVariation» **SwDataDefProps**			
swIsVirtual	Boolean	0..1	attr	This element distinguishes virtual objects. Virtual objects do not appear in the memory, their derivation is much more dependent on other objects and hence they shall have a swDataDependency . **Tags:** xml.sequenceOffset=260
swPointerTarget Props	SwPointerTargetProps	0..1	aggr	Specifies that the containing data object is a pointer to another data object. **Tags:** atp.Splitkey=swPointerTargetProps xml.sequenceOffset=280
swRecord Layout	SwRecordLayout	0..1	ref	Record layout for this data object. **Tags:** atp.Splitkey=swRecordLayout xml.sequenceOffset=290
swRefresh Timing	MultidimensionalTime	0..1	aggr	This element specifies the frequency in which the object involved shall be or is called or calculated. This timing can be collected from the task in which write access processes to the variable run. But this cannot be done by the MCD system. So this attribute can be used in an early phase to express the desired refresh timing and later on to specify the real refresh timing. **Tags:** atp.Splitkey=swRefreshTiming xml.sequenceOffset=300
swTextProps	SwTextProps	0..1	aggr	the specific properties if the data object is a text object. **Tags:** atp.Splitkey=swTextProps xml.sequenceOffset=120
swValueBlock Size	Numerical	0..1	attr	This represents the size of a Value Block **Stereotypes:** atpVariation **Tags:** vh.latestBindingTime=preCompileTime xml.sequenceOffset=80
swValueBlock SizeMult (ordered)	Numerical	*	attr	This attribute is used to specify the dimensions of a value block (VAL_BLK) for the case that that value block has more than one dimension. The dimensions given in this attribute are ordered such that the first entry represents the first dimension, the second entry represents the second dimension, and so on. For one-dimensional value blocks the attribute swValue BlockSize shall be used and this attribute shall not exist. **Stereotypes:** atpVariation **Tags:** vh.latestBindingTime=preCompileTime
unit	Unit	0..1	ref	Physical unit associated with the semantics of this data object. This attribute applies if no compuMethod is specified. If both units (this as well as via compuMethod) are specified the units shall be compatible. **Tags:** atp.Splitkey=unit xml.sequenceOffset=350

▽

△

Class	«atpVariation» **SwDataDefProps**			
valueAxisData Type	ApplicationPrimitive DataType	0..1	ref	The referenced ApplicationPrimitiveDataType represents the primitive data type of the value axis within a compound primitive (e.g. curve, map). It supersedes CompuMethod, Unit, and BaseType. **Tags:** atp.Splitkey=valueAxisDataType xml.sequenceOffset=355

Table A.35: SwDataDefProps

Class	**SymbolProps**			
Note	This meta-class represents the ability to contribute a part of a namespace.			
Base	*ARObject*, *ImplementationProps*, *Referrable*			
Aggregated by	Allocator.namespace, ApApplicationErrorDomain.namespace, *AtomicSwComponentType*.symbolProps, *CppImplementationDataType*.namespace, ImplementationDataType.symbolProps, *Port Interface*.namespace, SecurityEventDefinition.eventSymbolName, TraceSwitchConfig.namespace			
Attribute	*Type*	*Mult.*	*Kind*	*Note*
–	–	–	–	–

Table A.36: SymbolProps

Class	**UserDefinedEventDeployment**			
Note	UserDefined configuration settings for an Event. This Class is only used by the AUTOSAR Adaptive Platform.			
Base	*ARObject*, *Identifiable*, *MultilanguageReferrable*, *Referrable*, *ServiceEventDeployment*, *ServiceInterface DeploymentElement*			
Aggregated by	*ServiceInterfaceDeployment*.eventDeployment, UserDefinedFieldDeployment.notifier			
Attribute	*Type*	*Mult.*	*Kind*	*Note*
–	–	–	–	–

Table A.37: UserDefinedEventDeployment

Class	**UserDefinedFieldDeployment**			
Note	UserDefined configuration settings for a Field. This Class is only used by the AUTOSAR Adaptive Platform.			
Base	*ARObject*, *Identifiable*, *MultilanguageReferrable*, *Referrable*, *ServiceFieldDeployment*, *ServiceInterface DeploymentElement*			
Aggregated by	*ServiceInterfaceDeployment*.fieldDeployment			
Attribute	*Type*	*Mult.*	*Kind*	*Note*
get	UserDefinedMethod Deployment	0..1	aggr	This aggregation represents the settings of the get method **Tags:** atp.Splitkey=get, get.shortName
notifier	UserDefinedEvent Deployment	0..1	aggr	This aggregation represents the settings of the notifier. **Tags:** atp.Splitkey=notifier, notifier.shortName
set	UserDefinedMethod Deployment	0..1	aggr	This aggregation represents the settings of the set method **Tags:** atp.Splitkey=set, set.shortName

Table A.38: UserDefinedFieldDeployment

Class	**UserDefinedMethodDeployment**			
Note	UserDefined configuration settings for a Method. This Class is only used by the AUTOSAR Adaptive Platform.			
Base	*ARObject*, *Identifiable*, *MultilanguageReferrable*, *Referrable*, *ServiceInterfaceDeploymentElement*, *ServiceMethodDeployment*			
Aggregated by	*ServiceInterfaceDeployment*.methodDeployment, UserDefinedFieldDeployment.get, UserDefinedField Deployment.set			
Attribute	***Type***	***Mult.***	***Kind***	***Note***
–	–	–	–	–

Table A.39: UserDefinedMethodDeployment

Class	**UserDefinedServiceInterfaceDeployment**			
Note	UserDefined configuration settings for a ServiceInterface. **Tags:** atp.recommendedPackage=ServiceInterfaceDeployments This Class is only used by the AUTOSAR Adaptive Platform.			
Base	*ARElement*, *ARObject*, *CollectableElement*, *Identifiable*, *MultilanguageReferrable*, *PackageableElement*, *Referrable*, *ServiceInterfaceDeployment*, *UploadableDesignElement*, *UploadablePackageElement*			
Aggregated by	ARPackage.element			
Attribute	***Type***	***Mult.***	***Kind***	***Note***
–	–	–	–	–

Table A.40: UserDefinedServiceInterfaceDeployment

A.2 Examples (informative)

A.2.1 Root Package Structure

This example outlines to recommended root package structure

```
<?xml version='1.0' encoding='UTF-8'?>
<AUTOSAR xmlns="http://autosar.org/schema/r4.0"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation=
      "http://autosar.org/schema/r4.0 AUTOSAR_00055.xsd">
  <AR-PACKAGES>
    <AR-PACKAGE>
      <SHORT-NAME>Catalogs</SHORT-NAME>
      <ELEMENTS/>
      <AR-PACKAGES>
        <AR-PACKAGE>
          <SHORT-NAME>VSS</SHORT-NAME>
          <ELEMENTS/>
          <AR-PACKAGES>
            <AR-PACKAGE>
              <SHORT-NAME>ApplicationDataTypes</SHORT-NAME>
            </AR-PACKAGE>
            <AR-PACKAGE>
              <SHORT-NAME>CppImplementationDataTypes</SHORT-NAME>
            </AR-PACKAGE>
            <AR-PACKAGE>
              <SHORT-NAME>CompuMethods</SHORT-NAME>
            </AR-PACKAGE>
```

```
            <AR-PACKAGE>
              <SHORT-NAME>DataTypeMappings</SHORT-NAME>
            </AR-PACKAGE>
            <AR-PACKAGE>
              <SHORT-NAME>Units</SHORT-NAME>
            </AR-PACKAGE>
          </AR-PACKAGES>
        </AR-PACKAGE>
      </AR-PACKAGES>
    </AR-PACKAGE>
  </AR-PACKAGES>
</AUTOSAR>
```

Listing A.1: AUTOSAR ARXML representation of the root package structure

```
Vehicle:
  type: branch
  description: High-level vehicle data.

Vehicle.Cabin:
  type: branch
  description: Cabin data.

Vehicle.Cabin.Door:
  type: branch
  description: Door data.
```

This example shows the use of multiple catalog versions.

```
<?xml version='1.0' encoding='UTF-8'?>
<AUTOSAR xmlns="http://autosar.org/schema/r4.0"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation=
      "http://autosar.org/schema/r4.0 AUTOSAR_00055.xsd">
  <AR-PACKAGES>
    <AR-PACKAGE>
      <SHORT-NAME>Catalogs</SHORT-NAME>
      <ELEMENTS/>
      <AR-PACKAGES>
        <AR-PACKAGE>
          <SHORT-NAME>VSS</SHORT-NAME>
          <ELEMENTS/>
        </AR-PACKAGE>
        <AR-PACKAGE>
          <SHORT-NAME>VSS_4_1</SHORT-NAME>
          <ELEMENTS/>
        </AR-PACKAGE>
      </AR-PACKAGES>
    </AR-PACKAGE>
  </AR-PACKAGES>
</AUTOSAR>
```

Listing A.2: AUTOSAR ARXML Versioned Catalogs

This example outlines the resulting ARXML Package Structure

```
<?xml version='1.0' encoding='UTF-8'?>
<AUTOSAR xmlns="http://autosar.org/schema/r4.0" xmlns:xsi="http://www.w3.
    org/2001/XMLSchema-instance" xsi:schemaLocation="http://autosar.org/
    schema/r4.0 AUTOSAR_00055.xsd">
<AR-PACKAGES>
<AR-PACKAGE>
```

```
  <SHORT-NAME>VSS</SHORT-NAME>
  <ELEMENTS>
  </ELEMENTS>
  <AR-PACKAGES>
  <AR-PACKAGE>
    <SHORT-NAME>Vehicle</SHORT-NAME>
    <ELEMENTS/>
    <AR-PACKAGES>
    <AR-PACKAGE>
      <SHORT-NAME>Cabin</SHORT-NAME>
      <ELEMENTS/>
      <AR-PACKAGES>
      <AR-PACKAGE>
        <SHORT-NAME>Door</SHORT-NAME>
        <ELEMENTS/>
        <AR-PACKAGES/>
      </AR-PACKAGE>
      </AR-PACKAGES>
    </AR-PACKAGE>
    </AR-PACKAGES>
  </AR-PACKAGE>
  </AR-PACKAGES>
</AR-PACKAGE>
</AR-PACKAGES>
</AUTOSAR>
```

Listing A.3: AUTOSAR ARXML Branch Package Structure

A.2.2 Primitive Type Modelling (informative)

This example outlines the modelling for DataTypeMapping.

```
<?xml version='1.0' encoding='UTF-8'?>
<AUTOSAR xmlns="http://autosar.org/schema/r4.0" xmlns:xsi="http://www.w3.
    org/2001/XMLSchema-instance" xsi:schemaLocation="http://autosar.org/
    schema/r4.0 AUTOSAR_00055.xsd">
<AR-PACKAGES>
    <AR-PACKAGE>
        <SHORT-NAME>Catalogs</SHORT-NAME>
        <ELEMENTS/>
        <AR-PACKAGES>
            <AR-PACKAGE>
                <SHORT-NAME>VSS</SHORT-NAME>
                <ELEMENTS/>
                <AR-PACKAGES>
                    <AR-PACKAGE>
                        <SHORT-NAME>DataTypeMappings</SHORT-NAME>
                        <ELEMENTS>
                        <DATA-TYPE-MAPPING-SET>
                            <SHORT-NAME>DataTypeMappingSet</SHORT-NAME>
                            <DATA-TYPE-MAPS>
                            <DATA-TYPE-MAP>
                                <APPLICATION-DATA-TYPE-REF DEST="
                                    APPLICATION-PRIMITIVE-DATA-TYPE">/
```

```
                    Catalogs/VSS/ApplicationDataTypes/
                    Vehicle/NominalVoltage</APPLICATION-DATA
                    -TYPE-REF>
                <IMPLEMENTATION-DATA-TYPE-REF DEST="STD-CPP
                    -IMPLEMENTATION-DATA-TYPE">/Catalogs/VSS
                    /ImplementationDataTypes/Vehicle/
                    NominalVoltage</IMPLEMENTATION-DATA-TYPE
                    -REF>
            </DATA-TYPE-MAP>
            </DATA-TYPE-MAPS>
        </DATA-TYPE-MAPPING-SET>
        </ELEMENTS>
    </AR-PACKAGE>
    <AR-PACKAGE>
        <SHORT-NAME>ApplicationDataTypes</SHORT-NAME>
        <ELEMENTS>
        </ELEMENTS>
        <AR-PACKAGES>
            <AR-PACKAGE>
                <SHORT-NAME>Vehicle</SHORT-NAME>
                <ELEMENTS>
                    <APPLICATION-PRIMITIVE-DATA-TYPE>
                        <SHORT-NAME>NominalVoltage</SHORT-
                            NAME>
                        <DESC>
                            <L-2 L="EN">Nominal Voltage of
                                the battery.</L-2>
                        </DESC>
                        <CATEGORY>VALUE</CATEGORY>
                        <SW-DATA-DEF-PROPS>
                            <SW-DATA-DEF-PROPS-VARIANTS/>
                        </SW-DATA-DEF-PROPS>
                    </APPLICATION-PRIMITIVE-DATA-TYPE>
                </ELEMENTS>
            </AR-PACKAGE>
        </AR-PACKAGES>
    </AR-PACKAGE>
    <AR-PACKAGE>
        <SHORT-NAME>ImplementationDataTypes</SHORT-NAME>
        <ELEMENTS>
        </ELEMENTS>
        <AR-PACKAGES>
            <AR-PACKAGE>
                <SHORT-NAME>Vehicle</SHORT-NAME>
                <ELEMENTS>
                    <STD-CPP-IMPLEMENTATION-DATA-TYPE>
                    <SHORT-NAME>NominalVoltage</SHORT-NAME>
                    <CATEGORY>TYPE_REFERENCE</CATEGORY>
                    <NAMESPACES>
                        <SYMBOL-PROPS>
                        <SHORT-NAME>ns0</SHORT-NAME>
                        <SYMBOL>covesa</SYMBOL>
                        </SYMBOL-PROPS>
                        <SYMBOL-PROPS>
                        <SHORT-NAME>ns1</SHORT-NAME>
                        <SYMBOL>vss</SYMBOL>
```

```
                                        </SYMBOL-PROPS>
                                        <SYMBOL-PROPS>
                                        <SHORT-NAME>ns2</SHORT-NAME>
                                        <SYMBOL>vehicle</SYMBOL>
                                        </SYMBOL-PROPS>
                                        <SYMBOL-PROPS>
                                        <SHORT-NAME>ns3</SHORT-NAME>
                                        <SYMBOL>powertrain</SYMBOL>
                                        </SYMBOL-PROPS>
                                        <SYMBOL-PROPS>
                                        <SHORT-NAME>ns4</SHORT-NAME>
                                        <SYMBOL>tractionbattery</SYMBOL>
                                        </SYMBOL-PROPS>
                                    </NAMESPACES>
                                    <TYPE-EMITTER>TYPE_EMITTER_ARA</TYPE-
                                        EMITTER>
                                    <TYPE-REFERENCE-REF DEST="STD-CPP-
                                        IMPLEMENTATION-DATA-TYPE">/AUTOSAR/
                                        StdTypes/uint16_t</TYPE-REFERENCE-
                                        REF>
                                    </STD-CPP-IMPLEMENTATION-DATA-TYPE>
                                </ELEMENTS>
                            </AR-PACKAGE>
                        </AR-PACKAGES>
                    </AR-PACKAGE>
                </AR-PACKAGES>
            </AR-PACKAGE>
    </AR-PACKAGES>
    </AR-PACKAGE>
    <AR-PACKAGE>
      <SHORT-NAME>AUTOSAR</SHORT-NAME>
      <ELEMENTS/>
      <AR-PACKAGES>
        <AR-PACKAGE UUID="68d89532-bd07-4f66-907b-3b7e50daa1f8">
          <SHORT-NAME>StdTypes</SHORT-NAME>
          <ELEMENTS>
            <STD-CPP-IMPLEMENTATION-DATA-TYPE>
              <SHORT-NAME>uint16_t</SHORT-NAME>
              <CATEGORY>VALUE</CATEGORY>
            </STD-CPP-IMPLEMENTATION-DATA-TYPE>
          </ELEMENTS>
        </AR-PACKAGE>
      </AR-PACKAGES>
    </AR-PACKAGE>
</AR-PACKAGES>
</AUTOSAR>
```

Listing A.4: AUTOSAR ARXML Data Type Mapping

This example outlines the namespace modelling for StdCppImplementationDataType.

```
<?xml version='1.0' encoding='UTF-8'?>
<AUTOSAR xmlns="http://autosar.org/schema/r4.0"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xsi:schemaLocation="http://autosar.org/schema/r4.0 AUTOSAR_00055.xsd
       ">
    <AR-PACKAGES>
```

```
        <AR-PACKAGE>
            <SHORT-NAME>Door</SHORT-NAME>
            <ELEMENTS>
                <STD-CPP-IMPLEMENTATION-DATA-TYPE>
                    <SHORT-NAME>isLocked</SHORT-NAME>
                    <CATEGORY>TYPE_REFERENCE</CATEGORY>
                    <NAMESPACES>
                        <SYMBOL-PROPS>
                            <SHORT-NAME>ns0</SHORT-NAME>
                            <SYMBOL>covesa</SYMBOL>
                        </SYMBOL-PROPS>
                        <SYMBOL-PROPS>
                            <SHORT-NAME>ns1</SHORT-NAME>
                            <SYMBOL>vss</SYMBOL>
                        </SYMBOL-PROPS>
                        <SYMBOL-PROPS>
                            <SHORT-NAME>ns2</SHORT-NAME>
                            <SYMBOL>vehicle</SYMBOL>
                        </SYMBOL-PROPS>
                        <SYMBOL-PROPS>
                            <SHORT-NAME>ns3</SHORT-NAME>
                            <SYMBOL>cabin</SYMBOL>
                        </SYMBOL-PROPS>
                        <SYMBOL-PROPS>
                            <SHORT-NAME>ns4</SHORT-NAME>
                            <SYMBOL>door</SYMBOL>
                        </SYMBOL-PROPS>
                    </NAMESPACES>
                    <TYPE-EMITTER>TYPE_EMITTER_ARA</TYPE-EMITTER>
                    <TYPE-REFERENCE-REF DEST="STD-CPP-IMPLEMENTATION-DATA-
                        TYPE">/AUTOSAR/StdTypes/bool</TYPE-REFERENCE-REF>
                </STD-CPP-IMPLEMENTATION-DATA-TYPE>
            </ELEMENTS>
            <AR-PACKAGES/>
        </AR-PACKAGE>
        <AR-PACKAGE>
            <SHORT-NAME>AUTOSAR</SHORT-NAME>
            <ELEMENTS/>
            <AR-PACKAGES>
                <AR-PACKAGE UUID="68d89532-bd07-4f66-907b-3b7e50daa1f8">
                    <SHORT-NAME>StdTypes</SHORT-NAME>
                    <ELEMENTS>
                        <STD-CPP-IMPLEMENTATION-DATA-TYPE>
                            <SHORT-NAME>bool</SHORT-NAME>
                            <CATEGORY>VALUE</CATEGORY>
                        </STD-CPP-IMPLEMENTATION-DATA-TYPE>
                    </ELEMENTS>
                </AR-PACKAGE>
            </AR-PACKAGES>
        </AR-PACKAGE>
    </AR-PACKAGES>
</AUTOSAR>
```

Listing A.5: AUTOSAR ARXML Namespaces

A.2.3 C++ Keywords

The following items are currently C++ keywords:

- alignas (C++11)
- alignof (C++11)
- and
- and_eq
- asm
- atomic_cancel (TM TS)
- atomic_commit (TM TS)
- atomic_noexcept (TM TS)
- "auto"
- "bitand"
- bitor
- bool
- break
- case
- catch
- char
- char8_t (C++20)
- char16_t (C++11)
- char32_t (C++11)
- class
- compl
- concept (C++20)
- const
- consteval (C++20)
- constexpr (C++11)
- constinit (C++20)
- const_cast
- continue

- co_await (C++20)
- co_return (C++20)
- co_yield (C++20)
- decltype (C++11)
- default
- delete
- do
- double
- dynamic_cast
- else
- enum
- explicit
- export
- extern
- false
- float
- for
- friend
- goto
- if
- inline
- int
- long
- mutable
- namespace
- new
- noexcept (C++11)
- not
- not_eq
- nullptr (C++11)

- operator
- or
- or_eq
- private
- protected
- public
- reflexpr
- register
- reinterpret_cast
- requires (C++20)
- return
- short
- signed
- sizeof
- static
- static_assert (C++11)
- static_cast
- struct
- switch
- synchronized
- template
- this
- thread_local (C++11)
- throw
- true
- try
- typedef
- typeid
- typename
- union

- unsigned
- using
- virtual
- void
- volatile
- wchar_t
- while
- xor
- xor_eq

This example outlines the data constraint modelling

```
<?xml version='1.0' encoding='UTF-8'?>
<AUTOSAR xmlns="http://autosar.org/schema/r4.0" xmlns:xsi="http://www.w3.
    org/2001/XMLSchema-instance" xsi:schemaLocation="http://autosar.org/
    schema/r4.0 AUTOSAR_00055.xsd">
<AR-PACKAGES>
    <AR-PACKAGE>
        <SHORT-NAME>Catalogs</SHORT-NAME>
        <ELEMENTS/>
        <AR-PACKAGES>
            <AR-PACKAGE>
                <SHORT-NAME>VSS</SHORT-NAME>
                <ELEMENTS/>
                <AR-PACKAGES>
                    <AR-PACKAGE>
                    <SHORT-NAME>ApplicationDataTypes</SHORT-NAME>
                        <ELEMENTS>
                            <APPLICATION-PRIMITIVE-DATA-TYPE>
                            <SHORT-NAME>NominalVoltage</SHORT-NAME>
                            <DESC>
                            <L-2 L="EN">Nominal Voltage of the battery.</L
                                -2>
                            </DESC>
                            <CATEGORY>VALUE</CATEGORY>
                            <SW-DATA-DEF-PROPS>
                            <SW-DATA-DEF-PROPS-VARIANTS>
                                <SW-DATA-DEF-PROPS-CONDITIONAL>
                                <DATA-CONSTR-REF DEST="DATA-CONSTR">/
                                    Catalogs/VSS/ApplicationDataTypes/
                                    NominalVoltageDC</DATA-CONSTR-REF>
                                </SW-DATA-DEF-PROPS-CONDITIONAL>
                            </SW-DATA-DEF-PROPS-VARIANTS>
                            </SW-DATA-DEF-PROPS>
                            </APPLICATION-PRIMITIVE-DATA-TYPE>
                        <DATA-CONSTR>
                            <SHORT-NAME>NominalVoltageDC</SHORT-NAME>
                            <DATA-CONSTR-RULES>
                            <DATA-CONSTR-RULE>
```

```
                                    <PHYS-CONSTRS>
                                    <LOWER-LIMIT INTERVAL-TYPE="CLOSED">0</
                                        LOWER-LIMIT>
                                    <UPPER-LIMIT INTERVAL-TYPE="CLOSED">65535</
                                        UPPER-LIMIT>
                                    </PHYS-CONSTRS>
                                </DATA-CONSTR-RULE>
                                </DATA-CONSTR-RULES>
                            </DATA-CONSTR>
                            </ELEMENTS>
                            <AR-PACKAGES>
                            </AR-PACKAGES>
                        </AR-PACKAGE>
                    </AR-PACKAGES>
                </AR-PACKAGE>
    </AR-PACKAGES>
    </AR-PACKAGE>
</AR-PACKAGES>
</AUTOSAR>
```

Listing A.6: AUTOSAR ARXML data constraint

```
Vehicle:
  type: branch
  description: High-level vehicle data.

Vehicle.NominalVoltage:
    datatype: uint16[]
    type: attribute
    unit: V
    description: Nominal Voltage of the battery.
    comment: Nominal voltage typically refers to voltage of fully
      charged battery when delivering rated capacity.


Vehicle.NominalVoltage:
  datatype: uint16
  type: attribute
  unit: V
  description: Nominal Voltage of the battery.
  comment: Nominal voltage typically refers to voltage of fully
    charged battery when delivering rated capacity.


  SupportedAutonomyLevel:
  datatype: string
  type: attribute
  allowed: [
    'SAE_0', # No Driving Automation
    'SAE_1', # Driver Assistance
    'SAE_2', # Partial Driving Automation
    'SAE_3', # Conditional Driving Automation
    'SAE_4', # High Driving Automation
    'SAE_5'  # Full Driving Automation
  ]
  description: Indicates the highest level of driving automation
   according to the SAE J3016 taxonomy the vehicle is capable of.
```

```
<?xml version="1.0" encoding="UTF-8"?>
<AUTOSAR xmlns="http://autosar.org/schema/r4.0" xmlns:xsi="http://www.w3.
    org/2001/XMLSchema-instance" xsi:schemaLocation="http://autosar.org/
    schema/r4.0 AUTOSAR_00055.xsd">
  <AR-PACKAGES>
    <AR-PACKAGE>
      <SHORT-NAME>Catalogs</SHORT-NAME>
      <ELEMENTS />
      <AR-PACKAGES>
        <AR-PACKAGE>
          <SHORT-NAME>VSS</SHORT-NAME>
          <ELEMENTS />
          <AR-PACKAGES>
```

```
<AR-PACKAGE>
  <SHORT-NAME>ApplicationDataTypes</SHORT-NAME>
  <ELEMENTS>
    <APPLICATION-PRIMITIVE-DATA-TYPE>
      <SHORT-NAME>SupportedAutonomyLevel</SHORT-NAME>
      <DESC>
        <L-2 L="EN">Indicates the highest level of autonomy
            according to SAE J3016 taxonomy the vehicle is
            capable of.</L-2>
      </DESC>
      <CATEGORY>VALUE</CATEGORY>
      <SW-DATA-DEF-PROPS>
        <SW-DATA-DEF-PROPS-VARIANTS>
          <SW-DATA-DEF-PROPS-CONDITIONAL>
            <COMPU-METHOD-REF DEST="COMPU-METHOD">/Catalogs/VSS
                /ApplicationDataTypes/SupportedAutonomyLevelCM</
                COMPU-METHOD-REF>
          </SW-DATA-DEF-PROPS-CONDITIONAL>
        </SW-DATA-DEF-PROPS-VARIANTS>
      </SW-DATA-DEF-PROPS>
    </APPLICATION-PRIMITIVE-DATA-TYPE>
    <COMPU-METHOD>
      <SHORT-NAME>SupportedAutonomyLevelCM</SHORT-NAME>
      <CATEGORY>TEXTTABLE</CATEGORY>
      <COMPU-INTERNAL-TO-PHYS>
        <COMPU-SCALES>
          <COMPU-SCALE>
            <SYMBOL>SAE_0</SYMBOL>
            <LOWER-LIMIT>0</LOWER-LIMIT>
            <UPPER-LIMIT>0</UPPER-LIMIT>
          </COMPU-SCALE>
          <COMPU-SCALE>
            <SYMBOL>SAE_1</SYMBOL>
            <LOWER-LIMIT>1</LOWER-LIMIT>
            <UPPER-LIMIT>1</UPPER-LIMIT>
          </COMPU-SCALE>
          <COMPU-SCALE>
            <SYMBOL>SAE_2</SYMBOL>
            <LOWER-LIMIT>2</LOWER-LIMIT>
            <UPPER-LIMIT>2</UPPER-LIMIT>
          </COMPU-SCALE>
          <COMPU-SCALE>
            <SYMBOL>SAE_3</SYMBOL>
            <LOWER-LIMIT>3</LOWER-LIMIT>
            <UPPER-LIMIT>3</UPPER-LIMIT>
          </COMPU-SCALE>
          <COMPU-SCALE>
            <SYMBOL>SAE_4</SYMBOL>
            <LOWER-LIMIT>4</LOWER-LIMIT>
            <UPPER-LIMIT>4</UPPER-LIMIT>
          </COMPU-SCALE>
          <COMPU-SCALE>
            <SYMBOL>SAE_5</SYMBOL>
            <LOWER-LIMIT>5</LOWER-LIMIT>
            <UPPER-LIMIT>5</UPPER-LIMIT>
          </COMPU-SCALE>
```

```
                    </COMPU-SCALES>
                  </COMPU-INTERNAL-TO-PHYS>
                </COMPU-METHOD>
              </ELEMENTS>
            </AR-PACKAGE>
          </AR-PACKAGES>
        </AR-PACKAGE>
      </AR-PACKAGES>
    </AR-PACKAGE>
  </AR-PACKAGES>
</AUTOSAR>
```

Listing A.7: AUTOSAR ARXML representation of Enumeration

A.2.4 Complex Type Modelling

A.2.4.1 Variable Size Arrays

The following sections outlines a variable size array in VSS.

```
#
# The vehicle branch for highlevel vehicle signals
# and attributes.
#
Vehicle:
  type: branch
  description: High-level vehicle data.

# Include the Vehicle/Vehicle.vspec file and attach
# all its signals under the
# Vehicle branch created above.

Vehicle.NominalVoltage:
    datatype: uint16[]
    type: attribute
    unit: V
    description: Nominal Voltage of the battery.
    comment: Nominal voltage typically refers to voltage
      of fully charged
      battery when delivering rated capacity.
```

The resulting ARXML representations from that VSS array.

```
<?xml version="1.0" encoding="UTF-8"?>
<AUTOSAR xmlns="http://autosar.org/schema/r4.0" xmlns:xsi="http://www.w3.
    org/2001/XMLSchema-instance" xsi:schemaLocation="http://autosar.org/
    schema/r4.0 AUTOSAR_00055.xsd">
  <AR-PACKAGES>
    <AR-PACKAGE>
      <SHORT-NAME>Catalogs</SHORT-NAME>
      <ELEMENTS />
      <AR-PACKAGES>
        <AR-PACKAGE>
          <SHORT-NAME>VSS</SHORT-NAME>
          <ELEMENTS />
          <AR-PACKAGES>
            <AR-PACKAGE>
              <SHORT-NAME>ApplicationDataTypes</SHORT-NAME>
              <ELEMENTS />
              <AR-PACKAGES>
                <AR-PACKAGE>
                  <SHORT-NAME>Vehicle</SHORT-NAME>
                  <ELEMENTS>
```

```
                    <APPLICATION-PRIMITIVE-DATA-TYPE>
                      <SHORT-NAME>NominalVoltage</SHORT-NAME>
                      <DESC>
                        <L-2 L="EN">Nominal Voltage of the battery.</L-2>
                      </DESC>
                      <CATEGORY>VALUE</CATEGORY>
                    </APPLICATION-PRIMITIVE-DATA-TYPE>
                    <APPLICATION-ARRAY-DATA-TYPE>
                      <SHORT-NAME>NominalVoltage_ARRAY</SHORT-NAME>
                      <CATEGORY>ARRAY</CATEGORY>
                      <ELEMENT>
                        <SHORT-NAME>ARRAY_ELEMENT</SHORT-NAME>
                        <CATEGORY>VALUE</CATEGORY>
                        <TYPE-TREF DEST="APPLICATION-PRIMITIVE-DATA-TYPE">/
                            Catalogs/VSS/ApplicationDataTypes/Vehicle/
                            NominalVoltage</TYPE-TREF>
                        <ARRAY-SIZE-SEMANTICS>VARIABLE-SIZE</ARRAY-SIZE-
                            SEMANTICS>
                      </ELEMENT>
                    </APPLICATION-ARRAY-DATA-TYPE>
                  </ELEMENTS>
                </AR-PACKAGE>
              </AR-PACKAGES>
            </AR-PACKAGE>
          </AR-PACKAGES>
        </AR-PACKAGE>
      </AR-PACKAGES>
    </AR-PACKAGE>
  </AR-PACKAGES>
</AUTOSAR>
```

Listing A.8: AUTOSAR ARXML representation of the Variable Size Application Array Data Type

```
<?xml version="1.0" encoding="UTF-8"?>
<AUTOSAR xmlns="http://autosar.org/schema/r4.0" xmlns:xsi="http://www.w3.
    org/2001/XMLSchema-instance" xsi:schemaLocation="http://autosar.org/
    schema/r4.0 AUTOSAR_00055.xsd">
  <AR-PACKAGES>
    <AR-PACKAGE>
      <SHORT-NAME>Catalogs</SHORT-NAME>
      <ELEMENTS />
      <AR-PACKAGES>
        <AR-PACKAGE>
          <SHORT-NAME>VSS</SHORT-NAME>
          <ELEMENTS />
          <AR-PACKAGES>
            <AR-PACKAGE>
              <SHORT-NAME>ImplementationDataTypes</SHORT-NAME>
              <ELEMENTS>
                <STD-CPP-IMPLEMENTATION-DATA-TYPE>
                  <SHORT-NAME>NominalVoltage</SHORT-NAME>
                  <CATEGORY>TYPE_REFERENCE</CATEGORY>
                  <NAMESPACES>
                    <SYMBOL-PROPS>
                      <SHORT-NAME>ns0</SHORT-NAME>
```

```
                    <SYMBOL>covesa</SYMBOL>
                  </SYMBOL-PROPS>
                  <SYMBOL-PROPS>
                    <SHORT-NAME>ns1</SHORT-NAME>
                    <SYMBOL>vss</SYMBOL>
                  </SYMBOL-PROPS>
                </NAMESPACES>
                <TYPE-EMITTER>TYPE_EMITTER_ARA</TYPE-EMITTER>
                <TYPE-REFERENCE-REF DEST="STD-CPP-IMPLEMENTATION-DATA-
                    TYPE">/AUTOSAR/StdTypes/uint16_t</TYPE-REFERENCE-REF>
              </STD-CPP-IMPLEMENTATION-DATA-TYPE>
              <STD-CPP-IMPLEMENTATION-DATA-TYPE>
                <SHORT-NAME>NominalVoltage_ARRAY</SHORT-NAME>
                <CATEGORY>ARRAY</CATEGORY>
                <TEMPLATE-ARGUMENTS>
                  <CPP-TEMPLATE-ARGUMENT>
                    <TEMPLATE-TYPE-REF DEST="STD-CPP-IMPLEMENTATION-DATA-
                        TYPE">/Catalogs/VSS/ImplementationDataTypes/
                        NominalVoltage</TEMPLATE-TYPE-REF>
                  </CPP-TEMPLATE-ARGUMENT>
                </TEMPLATE-ARGUMENTS>
                <TYPE-EMITTER>TYPE_EMITTER_ARA</TYPE-EMITTER>
              </STD-CPP-IMPLEMENTATION-DATA-TYPE>
            </ELEMENTS>
          </AR-PACKAGE>
        </AR-PACKAGES>
      </AR-PACKAGE>
    </AR-PACKAGES>
  </AR-PACKAGE>
  <AR-PACKAGE>
    <SHORT-NAME>AUTOSAR</SHORT-NAME>
    <ELEMENTS />
    <AR-PACKAGES>
      <AR-PACKAGE UUID="68d89532-bd07-4f66-907b-3b7e50daa1f8">
        <SHORT-NAME>StdTypes</SHORT-NAME>
        <ELEMENTS>
          <STD-CPP-IMPLEMENTATION-DATA-TYPE>
            <SHORT-NAME>uint16_t</SHORT-NAME>
            <CATEGORY>VALUE</CATEGORY>
          </STD-CPP-IMPLEMENTATION-DATA-TYPE>
        </ELEMENTS>
      </AR-PACKAGE>
    </AR-PACKAGES>
  </AR-PACKAGE>
</AR-PACKAGES>
</AUTOSAR>
```

Listing A.9: AUTOSAR ARXML representation of the Variable Size Implementation Array Data Type

A.2.4.2 Fixed Size Arrays

```
Vehicle:
  type: branch
  description: High-level vehicle data.
```

```
Vehicle.NominalVoltage:
    datatype: uint16[]
    type: attribute
    unit: V
    description: Nominal Voltage of the battery.
    comment: Nominal voltage typically refers
       to voltage of fully charged
       battery when delivering rated capacity.
```

```
<?xml version="1.0" encoding="UTF-8"?>
<AUTOSAR xmlns="http://autosar.org/schema/r4.0" xmlns:xsi="http://www.w3.
    org/2001/XMLSchema-instance" xsi:schemaLocation="http://autosar.org/
    schema/r4.0 AUTOSAR_00055.xsd">
  <AR-PACKAGES>
    <AR-PACKAGE>
      <SHORT-NAME>Catalogs</SHORT-NAME>
      <ELEMENTS />
      <AR-PACKAGES>
        <AR-PACKAGE>
          <SHORT-NAME>VSS</SHORT-NAME>
          <ELEMENTS />
          <AR-PACKAGES>
            <AR-PACKAGE>
              <SHORT-NAME>ApplicationDataTypes</SHORT-NAME>
              <ELEMENTS />
              <AR-PACKAGES>
                <AR-PACKAGE>
                  <SHORT-NAME>Vehicle</SHORT-NAME>
                  <ELEMENTS>
                    <APPLICATION-PRIMITIVE-DATA-TYPE>
                      <SHORT-NAME>NominalVoltage</SHORT-NAME>
                      <DESC>
                        <L-2 L="EN">Nominal Voltage of the battery.</L-2>
                      </DESC>
                      <CATEGORY>VALUE</CATEGORY>
                    </APPLICATION-PRIMITIVE-DATA-TYPE>
                    <APPLICATION-ARRAY-DATA-TYPE>
                      <SHORT-NAME>NominalVoltage_ARRAY</SHORT-NAME>
                      <CATEGORY>ARRAY</CATEGORY>
                      <ELEMENT>
                        <SHORT-NAME>ARRAY_ELEMENT</SHORT-NAME>
                        <CATEGORY>VALUE</CATEGORY>
                        <TYPE-TREF DEST="APPLICATION-PRIMITIVE-DATA-TYPE">/
                            Catalogs/VSS/ApplicationDataTypes/Vehicle/
                            NominalVoltage</TYPE-TREF>
                        <ARRAY-SIZE-SEMANTICS>FIXED-SIZE</ARRAY-SIZE-
                            SEMANTICS>
                        <MAX-NUMBER-OF-ELEMENTS>10</MAX-NUMBER-OF-ELEMENTS>
                      </ELEMENT>
                    </APPLICATION-ARRAY-DATA-TYPE>
                  </ELEMENTS>
                </AR-PACKAGE>
              </AR-PACKAGES>
            </AR-PACKAGE>
          </AR-PACKAGES>
        </AR-PACKAGE>
      </AR-PACKAGES>
    </AR-PACKAGE>
```

```
  </AR-PACKAGES>
</AUTOSAR>
```

Listing A.10: AUTOSAR ARXML representation of the Fixed Size Application Array Data Type

```
<?xml version="1.0" encoding="UTF-8"?>
<AUTOSAR xmlns="http://autosar.org/schema/r4.0" xmlns:xsi="http://www.w3.
    org/2001/XMLSchema-instance" xsi:schemaLocation="http://autosar.org/
    schema/r4.0 AUTOSAR_00055.xsd">
  <AR-PACKAGES>
    <AR-PACKAGE>
      <SHORT-NAME>Catalogs</SHORT-NAME>
      <ELEMENTS />
      <AR-PACKAGES>
        <AR-PACKAGE>
          <SHORT-NAME>VSS</SHORT-NAME>
          <ELEMENTS />
          <AR-PACKAGES>
            <AR-PACKAGE>
              <SHORT-NAME>ImplementationDataTypes</SHORT-NAME>
              <ELEMENTS>
                <STD-CPP-IMPLEMENTATION-DATA-TYPE>
                  <SHORT-NAME>NominalVoltage</SHORT-NAME>
                  <CATEGORY>TYPE_REFERENCE</CATEGORY>
                  <NAMESPACES>
                    <SYMBOL-PROPS>
                      <SHORT-NAME>ns0</SHORT-NAME>
                      <SYMBOL>covesa</SYMBOL>
                    </SYMBOL-PROPS>
                    <SYMBOL-PROPS>
                      <SHORT-NAME>ns1</SHORT-NAME>
                      <SYMBOL>vss</SYMBOL>
                    </SYMBOL-PROPS>
                  </NAMESPACES>
                  <TYPE-EMITTER>TYPE_EMITTER_ARA</TYPE-EMITTER>
                  <TYPE-REFERENCE-REF DEST="STD-CPP-IMPLEMENTATION-DATA-
                      TYPE">/AUTOSAR/StdTypes/uint16_t</TYPE-REFERENCE-REF>
                </STD-CPP-IMPLEMENTATION-DATA-TYPE>
                <STD-CPP-IMPLEMENTATION-DATA-TYPE>
                  <SHORT-NAME>NominalVoltage_ARRAY</SHORT-NAME>
                  <CATEGORY>ARRAY</CATEGORY>
                  <ARRAY-SIZE>10</ARRAY-SIZE>
                  <TEMPLATE-ARGUMENTS>
                    <CPP-TEMPLATE-ARGUMENT>
                      <TEMPLATE-TYPE-REF DEST="STD-CPP-IMPLEMENTATION-DATA-
                          TYPE">/Catalogs/VSS/ImplementationDataTypes/
                          NominalVoltage</TEMPLATE-TYPE-REF>
                    </CPP-TEMPLATE-ARGUMENT>
                  </TEMPLATE-ARGUMENTS>
                  <TYPE-EMITTER>TYPE_EMITTER_ARA</TYPE-EMITTER>
                </STD-CPP-IMPLEMENTATION-DATA-TYPE>
              </ELEMENTS>
            </AR-PACKAGE>
          </AR-PACKAGES>
        </AR-PACKAGE>
```

```
      </AR-PACKAGES>
    </AR-PACKAGE>
    <AR-PACKAGE>
      <SHORT-NAME>AUTOSAR</SHORT-NAME>
      <ELEMENTS />
      <AR-PACKAGES>
        <AR-PACKAGE>
          <SHORT-NAME>StdTypes</SHORT-NAME>
          <ELEMENTS>
            <STD-CPP-IMPLEMENTATION-DATA-TYPE>
              <SHORT-NAME>uint16_t</SHORT-NAME>
              <CATEGORY>VALUE</CATEGORY>
            </STD-CPP-IMPLEMENTATION-DATA-TYPE>
          </ELEMENTS>
        </AR-PACKAGE>
      </AR-PACKAGES>
    </AR-PACKAGE>
  </AR-PACKAGES>
</AUTOSAR>
```

Listing A.11: AUTOSAR ARXML representation of the Fixed Size Implementation Array Data Type

A.2.4.3 Struct Types

```
Types:
  type: branch

Types.DeliveryInfo:
  type: struct
  description: A struct datatype containing info for each delivery

Types.DeliveryInfo.Address:
  datatype: string
  type: property
  description: Destination address

Types.DeliveryInfo.Receiver:
  datatype: string
  type: property
  description: Name of receiver
```

```
<?xml version="1.0" encoding="UTF-8"?>
<AUTOSAR xmlns="http://autosar.org/schema/r4.0" xmlns:xsi="http://www.w3.
    org/2001/XMLSchema-instance" xsi:schemaLocation="http://autosar.org/
    schema/r4.0 AUTOSAR_00055.xsd">
  <AR-PACKAGES>
    <AR-PACKAGE>
      <SHORT-NAME>Catalogs</SHORT-NAME>
      <ELEMENTS />
      <AR-PACKAGES>
        <AR-PACKAGE>
          <SHORT-NAME>VSS</SHORT-NAME>
          <ELEMENTS />
          <AR-PACKAGES>
            <AR-PACKAGE>
              <SHORT-NAME>ApplicationDataTypes</SHORT-NAME>
              <ELEMENTS />
              <AR-PACKAGES>
                <AR-PACKAGE>
                  <SHORT-NAME>Types</SHORT-NAME>
                  <ELEMENTS>
                    <APPLICATION-PRIMITIVE-DATA-TYPE>
```

```
                    <SHORT-NAME>Address</SHORT-NAME>
                    <DESC>
                      <L-2 L="EN">Destination address</L-2>
                    </DESC>
                    <CATEGORY>VALUE</CATEGORY>
                  </APPLICATION-PRIMITIVE-DATA-TYPE>
                  <APPLICATION-PRIMITIVE-DATA-TYPE>
                    <SHORT-NAME>Receiver</SHORT-NAME>
                    <DESC>
                      <L-2 L="EN">Name of receiver</L-2>
                    </DESC>
                    <CATEGORY>VALUE</CATEGORY>
                  </APPLICATION-PRIMITIVE-DATA-TYPE>
                  <APPLICATION-RECORD-DATA-TYPE>
                    <SHORT-NAME>DeliveryInfo</SHORT-NAME>
                    <DESC>
                      <L-2 L="EN">A struct datatype containing info for
                          each delivery</L-2>
                    </DESC>
                    <CATEGORY>STRUCTURE</CATEGORY>
                    <ELEMENTS>
                      <APPLICATION-RECORD-ELEMENT>
                        <SHORT-NAME>Address</SHORT-NAME>
                        <CATEGORY>VALUE</CATEGORY>
                        <TYPE-TREF DEST="APPLICATION-PRIMITIVE-DATA-TYPE"
                            >/Catalogs/VSS/ApplicationDataTypes/Types/
                            Address</TYPE-TREF>
                      </APPLICATION-RECORD-ELEMENT>
                      <APPLICATION-RECORD-ELEMENT>
                        <SHORT-NAME>Receiver</SHORT-NAME>
                        <CATEGORY>VALUE</CATEGORY>
                        <TYPE-TREF DEST="APPLICATION-PRIMITIVE-DATA-TYPE"
                            >/Catalogs/VSS/ApplicationDataTypes/Types/
                            Receiver</TYPE-TREF>
                      </APPLICATION-RECORD-ELEMENT>
                    </ELEMENTS>
                  </APPLICATION-RECORD-DATA-TYPE>
                </ELEMENTS>
                <AR-PACKAGES />
              </AR-PACKAGE>
            </AR-PACKAGES>
          </AR-PACKAGE>
        </AR-PACKAGES>
      </AR-PACKAGE>
    </AR-PACKAGES>
  </AR-PACKAGE>
</AR-PACKAGES>
</AUTOSAR>
```

Listing A.12: AUTOSAR ARXML representation of the Application Struct Data Type

```
<?xml version="1.0" encoding="UTF-8"?>
<AUTOSAR xmlns="http://autosar.org/schema/r4.0" xmlns:xsi="http://www.w3.
    org/2001/XMLSchema-instance" xsi:schemaLocation="http://autosar.org/
    schema/r4.0 AUTOSAR_00055.xsd">
  <AR-PACKAGES>
```

```
<AR-PACKAGE>
  <SHORT-NAME>Catalogs</SHORT-NAME>
  <ELEMENTS />
  <AR-PACKAGES>
    <AR-PACKAGE>
      <SHORT-NAME>VSS</SHORT-NAME>
      <ELEMENTS />
      <AR-PACKAGES>
        <AR-PACKAGE>
          <SHORT-NAME>ImplementationDataTypes</SHORT-NAME>
          <ELEMENTS>
            <STD-CPP-IMPLEMENTATION-DATA-TYPE>
              <SHORT-NAME>String</SHORT-NAME>
              <DESC>
                <L-2 L="EN">Dynamic string implementation.</L-2>
              </DESC>
              <CATEGORY>STRING</CATEGORY>
              <TYPE-EMITTER>TYPE_EMITTER_ARA</TYPE-EMITTER>
            </STD-CPP-IMPLEMENTATION-DATA-TYPE>
          </ELEMENTS>
          <AR-PACKAGES>
            <AR-PACKAGE>
              <SHORT-NAME>Types</SHORT-NAME>
              <ELEMENTS>
                <STD-CPP-IMPLEMENTATION-DATA-TYPE>
                  <SHORT-NAME>Receiver</SHORT-NAME>
                  <CATEGORY>TYPE_REFERENCE</CATEGORY>
                  <NAMESPACES>
                    <SYMBOL-PROPS>
                      <SHORT-NAME>ns0</SHORT-NAME>
                      <SYMBOL>covesa</SYMBOL>
                    </SYMBOL-PROPS>
                    <SYMBOL-PROPS>
                      <SHORT-NAME>ns1</SHORT-NAME>
                      <SYMBOL>vss</SYMBOL>
                    </SYMBOL-PROPS>
                  </NAMESPACES>
                  <TYPE-EMITTER>TYPE_EMITTER_ARA</TYPE-EMITTER>
                  <TYPE-REFERENCE-REF DEST="STD-CPP-IMPLEMENTATION-DATA
                      -TYPE">/Catalogs/VSS/ImplementationDataTypes/
                      String</TYPE-REFERENCE-REF>
                </STD-CPP-IMPLEMENTATION-DATA-TYPE>
                <STD-CPP-IMPLEMENTATION-DATA-TYPE>
                  <SHORT-NAME>Address</SHORT-NAME>
                  <CATEGORY>TYPE_REFERENCE</CATEGORY>
                  <NAMESPACES>
                    <SYMBOL-PROPS>
                      <SHORT-NAME>ns0</SHORT-NAME>
                      <SYMBOL>covesa</SYMBOL>
                    </SYMBOL-PROPS>
                    <SYMBOL-PROPS>
                      <SHORT-NAME>ns1</SHORT-NAME>
                      <SYMBOL>vss</SYMBOL>
                    </SYMBOL-PROPS>
                  </NAMESPACES>
                  <TYPE-EMITTER>TYPE_EMITTER_ARA</TYPE-EMITTER>
```

```
                    <TYPE-REFERENCE-REF DEST="STD-CPP-IMPLEMENTATION-DATA
                        -TYPE">/Catalogs/VSS/ImplementationDataTypes/
                        String</TYPE-REFERENCE-REF>
                  </STD-CPP-IMPLEMENTATION-DATA-TYPE>
                  <STD-CPP-IMPLEMENTATION-DATA-TYPE>
                    <SHORT-NAME>DeliveryInfo</SHORT-NAME>
                    <CATEGORY>STRUCTURE</CATEGORY>
                    <SUB-ELEMENTS>
                      <CPP-IMPLEMENTATION-DATA-TYPE-ELEMENT>
                        <SHORT-NAME>Address</SHORT-NAME>
                        <TYPE-REFERENCE>
                          <TYPE-REFERENCE-REF DEST="STD-CPP-
                              IMPLEMENTATION-DATA-TYPE">/Catalogs/VSS/
                              ImplementationDataTypes/Types/Address</TYPE-
                              REFERENCE-REF>
                        </TYPE-REFERENCE>
                      </CPP-IMPLEMENTATION-DATA-TYPE-ELEMENT>
                      <CPP-IMPLEMENTATION-DATA-TYPE-ELEMENT>
                        <SHORT-NAME>Receiver</SHORT-NAME>
                        <TYPE-REFERENCE>
                          <TYPE-REFERENCE-REF DEST="STD-CPP-
                              IMPLEMENTATION-DATA-TYPE">/Catalogs/VSS/
                              ImplementationDataTypes/Types/Receiver</TYPE
                              -REFERENCE-REF>
                        </TYPE-REFERENCE>
                      </CPP-IMPLEMENTATION-DATA-TYPE-ELEMENT>
                    </SUB-ELEMENTS>
                    <TYPE-EMITTER>TYPE_EMITTER_ARA</TYPE-EMITTER>
                  </STD-CPP-IMPLEMENTATION-DATA-TYPE>
                </ELEMENTS>
              </AR-PACKAGE>
            </AR-PACKAGES>
          </AR-PACKAGE>
        </AR-PACKAGES>
      </AR-PACKAGE>
    </AR-PACKAGES>
  </AR-PACKAGE>
</AUTOSAR>
```

Listing A.13: AUTOSAR ARXML representation of the Implementation Struct Data Type

```
<?xml version='1.0' encoding='UTF-8'?>
<AUTOSAR xmlns="http://autosar.org/schema/r4.0"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xsi:schemaLocation="http://autosar.org/schema/r4.0 AUTOSAR_00055.xsd
       ">
    <AR-PACKAGES>
        <AR-PACKAGE>
            <SHORT-NAME>VehicleIdentification</SHORT-NAME>
            <ELEMENTS>
                <APPLICATION-PRIMITIVE-DATA-TYPE>
                    <SHORT-NAME>VIN</SHORT-NAME>
                    <DESC>
                        <L-2 L="EN">17-character Vehicle Identification
                            Number (VIN) as defined by ISO 3779.</L-2>
```

```
                    </DESC>
                    <CATEGORY>STRING</CATEGORY>
                    <ADMIN-DATA>
                        <SDGS>
                            <SDG GID="VSS">
                                <SD GID="myVSSKey">myVSSValue</SD>
                                <SD GID="anotherVSSKey">anotherVSSValue</SD
                                    >
                            </SDG>
                        </SDGS>
                    </ADMIN-DATA>
                </APPLICATION-PRIMITIVE-DATA-TYPE>
            </ELEMENTS>
        </AR-PACKAGE>
    </AR-PACKAGES>
</AUTOSAR>
```

Listing A.14: AUTOSAR ARXML representation of arbitrary VSS node key value pairs

A.2.5 VISS Specific Model Extension

This is an example of the VISS specific model extensions. Note that model extensions are not part of the main arxml model file.

```
<?xml version="1.0" encoding="UTF-8"?>
<AUTOSAR xmlns="http://autosar.org/schema/r4.0" xmlns:xsi="http://www.w3.
    org/2001/XMLSchema-instance" xsi:schemaLocation="http://autosar.org/
    schema/r4.0 AUTOSAR_00055.xsd">
  <AR-PACKAGES>
    <AR-PACKAGE>
      <SHORT-NAME>Extensions</SHORT-NAME>
      <AR-PACKAGES>
        <AR-PACKAGE>
          <SHORT-NAME>ADAPTIVE</SHORT-NAME>
          <ELEMENTS>
            <SDG-DEF>
              <SHORT-NAME>UserDefinedVISSBindingExtensionModel</SHORT-NAME>
              <ADMIN-DATA>
                <SDGS>
                  <SDG GID="VISSMex-version">
                    <SD>1.0.0</SD>
                  </SDG>
                </SDGS>
              </ADMIN-DATA>
              <SDG-CLASSES>
                <!-- Provided VISS Service Instance -->
                <SDG-CLASS>
                  <SHORT-NAME>ProvidedVISSServiceInstanceExtension</SHORT-
                      NAME>
                  <EXTENDS-META-CLASS>ProvidedUserDefinedServiceInstance</
                      EXTENDS-META-CLASS>
                  <ATTRIBUTES>
                    <SDG-AGGREGATION-WITH-VARIATION>
                      <SHORT-NAME>userDefinedVISSBinding</SHORT-NAME>
                      <LOWER-MULTIPLICITY>0</LOWER-MULTIPLICITY>
                      <UPPER-MULTIPLICITY>1</UPPER-MULTIPLICITY>
```

```
        <VARIATION>false</VARIATION>
        <SUB-SDG-REF DEST="SDG-CLASS">/Extensions/ADAPTIVE/
            UserDefinedVISSBindingExtensionModel/
            ProvidedVISSServiceInstance</SUB-SDG-REF>
      </SDG-AGGREGATION-WITH-VARIATION>
    </ATTRIBUTES>
  </SDG-CLASS>
  <SDG-CLASS>
    <SHORT-NAME>ProvidedVISSServiceInstance</SHORT-NAME>
    <DESC>
      <L-2 L="FOR-ALL">Stores the VISS Binding data of the
          ProvidedUserDefinedServiceInstance</L-2>
    </DESC>
    <ATTRIBUTES>
      <SDG-PRIMITIVE-ATTRIBUTE>
        <SHORT-NAME>vssInstancePath</SHORT-NAME>
        <DESC>
          <L-2 L="FOR-ALL">The service instance vss path</L-2
              >
        </DESC>
        <CATEGORY>STRING</CATEGORY>
        <LOWER-MULTIPLICITY>0</LOWER-MULTIPLICITY>
        <UPPER-MULTIPLICITY>1</UPPER-MULTIPLICITY>
      </SDG-PRIMITIVE-ATTRIBUTE>
    </ATTRIBUTES>
  </SDG-CLASS>
  <SDG-CLASS>
    <SHORT-NAME>Version</SHORT-NAME>
    <ATTRIBUTES>
      <SDG-PRIMITIVE-ATTRIBUTE>
        <SHORT-NAME>major</SHORT-NAME>
        <DESC>
          <L-2 L="FOR-ALL">The major version (32 bit unsigned
              integer)</L-2>
        </DESC>
        <CATEGORY>INTEGER</CATEGORY>
        <LOWER-MULTIPLICITY>0</LOWER-MULTIPLICITY>
        <UPPER-MULTIPLICITY>1</UPPER-MULTIPLICITY>
        <MAX>0xFFFFFFFF</MAX>
        <MIN>0</MIN>
      </SDG-PRIMITIVE-ATTRIBUTE>
      <SDG-PRIMITIVE-ATTRIBUTE>
        <SHORT-NAME>minor</SHORT-NAME>
        <DESC>
          <L-2 L="FOR-ALL">The minor version (32 bit unsigned
              integer) of all offered service instances.
This minor version is only relevant for skeleton-side
    service offers.</L-2>
        </DESC>
        <CATEGORY>INTEGER</CATEGORY>
        <LOWER-MULTIPLICITY>0</LOWER-MULTIPLICITY>
        <UPPER-MULTIPLICITY>1</UPPER-MULTIPLICITY>
        <MAX>0xFFFFFFFE</MAX>
        <MIN>0</MIN>
      </SDG-PRIMITIVE-ATTRIBUTE>
    </ATTRIBUTES>
```

```
</SDG-CLASS>
<!-- VISS Service Interface Deployment -->
<SDG-CLASS>
  <SHORT-NAME>
     VISSBindingServiceInterfaceDeploymentExtension</SHORT-
     NAME>
  <EXTENDS-META-CLASS>UserDefinedServiceInterfaceDeployment
     </EXTENDS-META-CLASS>
  <ATTRIBUTES>
    <SDG-AGGREGATION-WITH-VARIATION>
      <SHORT-NAME>userDefinedVISSBinding</SHORT-NAME>
      <LOWER-MULTIPLICITY>0</LOWER-MULTIPLICITY>
      <UPPER-MULTIPLICITY>1</UPPER-MULTIPLICITY>
      <VARIATION>false</VARIATION>
      <SUB-SDG-REF DEST="SDG-CLASS">/Extensions/ADAPTIVE/
         UserDefinedVISSBindingExtensionModel/
         VISSBindingServiceInterfaceDeployment</SUB-SDG-REF
         >
    </SDG-AGGREGATION-WITH-VARIATION>
  </ATTRIBUTES>
</SDG-CLASS>
<SDG-CLASS>
  <SHORT-NAME>VISSBindingServiceInterfaceDeployment</SHORT-
     NAME>
  <DESC>
    <L-2 L="FOR-ALL">Stores the VISSBinding data of the
       UserDefinedServiceInterfaceDeployment</L-2>
  </DESC>
  <ATTRIBUTES>
    <SDG-PRIMITIVE-ATTRIBUTE>
      <SHORT-NAME>serviceInterfacePath</SHORT-NAME>
      <DESC>
        <L-2 L="FOR-ALL">The VSS type path of the service
           interface(string)</L-2>
      </DESC>
      <CATEGORY>INTEGER</CATEGORY>
      <LOWER-MULTIPLICITY>0</LOWER-MULTIPLICITY>
      <UPPER-MULTIPLICITY>1</UPPER-MULTIPLICITY>
      <MAX>0xFFFFFFFF</MAX>
      <MIN>0</MIN>
    </SDG-PRIMITIVE-ATTRIBUTE>
    <SDG-AGGREGATION-WITH-VARIATION>
      <SHORT-NAME>serviceInterfaceVersion</SHORT-NAME>
      <LOWER-MULTIPLICITY>0</LOWER-MULTIPLICITY>
      <UPPER-MULTIPLICITY>1</UPPER-MULTIPLICITY>
      <VARIATION>false</VARIATION>
      <SUB-SDG-REF DEST="SDG-CLASS">/Extensions/ADAPTIVE/
         UserDefinedVISSBindingExtensionModel/Version</SUB-
         SDG-REF>
    </SDG-AGGREGATION-WITH-VARIATION>
  </ATTRIBUTES>
</SDG-CLASS>
<SDG-CLASS>
  <SHORT-NAME>VISSBindingEventDeploymentExtension</SHORT-
     NAME>
```

```
                <EXTENDS-META-CLASS>UserDefinedEventDeployment</EXTENDS-
                    META-CLASS>
                <ATTRIBUTES>
                  <SDG-AGGREGATION-WITH-VARIATION>
                    <SHORT-NAME>userDefinedVISSBinding</SHORT-NAME>
                    <LOWER-MULTIPLICITY>0</LOWER-MULTIPLICITY>
                    <UPPER-MULTIPLICITY>1</UPPER-MULTIPLICITY>
                    <VARIATION>false</VARIATION>
                    <SUB-SDG-REF DEST="SDG-CLASS">/Extensions/ADAPTIVE/
                        UserDefinedVISSBindingExtensionModel/
                        VISSBindingEventDeployment</SUB-SDG-REF>
                  </SDG-AGGREGATION-WITH-VARIATION>
                </ATTRIBUTES>
              </SDG-CLASS>
              <SDG-CLASS>
                <SHORT-NAME>VISSBindingEventDeployment</SHORT-NAME>
                <DESC>
                  <L-2 L="FOR-ALL">Stores the VISSBinding data of the
                      UserDefinedEventDeployment</L-2>
                </DESC>
                <ATTRIBUTES />
              </SDG-CLASS>
              <SDG-CLASS>
                <SHORT-NAME>VISSBindingMethodDeploymentExtension</SHORT-
                    NAME>
                <EXTENDS-META-CLASS>UserDefinedMethodDeployment</EXTENDS-
                    META-CLASS>
                <ATTRIBUTES>
                  <SDG-AGGREGATION-WITH-VARIATION>
                    <SHORT-NAME>userDefinedVISSBinding</SHORT-NAME>
                    <LOWER-MULTIPLICITY>0</LOWER-MULTIPLICITY>
                    <UPPER-MULTIPLICITY>1</UPPER-MULTIPLICITY>
                    <VARIATION>false</VARIATION>
                    <SUB-SDG-REF DEST="SDG-CLASS">/Extensions/ADAPTIVE/
                        UserDefinedVISSBindingExtensionModel/
                        VISSBindingMethodDeployment</SUB-SDG-REF>
                  </SDG-AGGREGATION-WITH-VARIATION>
                </ATTRIBUTES>
              </SDG-CLASS>
              <SDG-CLASS>
                <SHORT-NAME>VISSBindingMethodDeployment</SHORT-NAME>
                <DESC>
                  <L-2 L="FOR-ALL">Stores the VISSBinding data of the
                      UserDefinedMethodDeployment</L-2>
                </DESC>
                <ATTRIBUTES />
              </SDG-CLASS>
            </SDG-CLASSES>
          </SDG-DEF>
        </ELEMENTS>
      </AR-PACKAGE>
    </AR-PACKAGES>
  </AR-PACKAGE>
</AR-PACKAGES>
```

```
</AUTOSAR>
```

Listing A.15: VISS Model Extension

A.2.6 User Defined VISS Service Deployments

```
<?xml version="1.0" encoding="UTF-8"?>
<AUTOSAR xmlns="http://autosar.org/schema/r4.0" xmlns:xsi="http://www.w3.
    org/2001/XMLSchema-instance" xsi:schemaLocation="http://autosar.org/
    schema/r4.0 AUTOSAR_00055.xsd">
  <AR-PACKAGES>
    <AR-PACKAGE>
      <SHORT-NAME>Catalogs</SHORT-NAME>
      <ELEMENTS />
      <AR-PACKAGES>
        <AR-PACKAGE>
          <SHORT-NAME>VSS</SHORT-NAME>
          <ELEMENTS />
          <AR-PACKAGES>
            <AR-PACKAGE>
              <SHORT-NAME>ServiceInterfaces</SHORT-NAME>
              <ELEMENTS />
              <AR-PACKAGES>
                <AR-PACKAGE>
                  <SHORT-NAME>Vehicle</SHORT-NAME>
                  <ELEMENTS>
                    <SERVICE-INTERFACE UUID="29987d97-5605-49a8-a378-
                        c6478d055292">
                      <SHORT-NAME>Vehicle</SHORT-NAME>
                      <FIELDS>
                        <FIELD UUID="17b45063-f713-4904-bb60-37969a7ce346">
                          <SHORT-NAME>LowVoltageSystemState</SHORT-NAME>
                          <!--<TYPE-TREF DEST="STD-CPP-IMPLEMENTATION-DATA-
                              TYPE">/Catalogues/ImplementationDataTypes/
                              VehicleSignalSpecification/Vehicle/
                              LowVoltageSystemState</TYPE-TREF>-->
                          <HAS-GETTER>true</HAS-GETTER>
                          <HAS-NOTIFIER>true</HAS-NOTIFIER>
                          <HAS-SETTER>true</HAS-SETTER>
                        </FIELD>
                      </FIELDS>
                    </SERVICE-INTERFACE>
                  </ELEMENTS>
                </AR-PACKAGE>
              </AR-PACKAGES>
            </AR-PACKAGE>
            <AR-PACKAGE>
              <SHORT-NAME>VISSIPServiceInterfaceDeployments</SHORT-NAME>
              <ELEMENTS />
              <AR-PACKAGES>
                <AR-PACKAGE>
                  <SHORT-NAME>Vehicle</SHORT-NAME>
                  <ELEMENTS>
                    <USER-DEFINED-SERVICE-INTERFACE-DEPLOYMENT>
```

```
<SHORT-NAME>Vehicle</SHORT-NAME>
<ADMIN-DATA>
  <SDGS>
    <SDG GID="
       VISSMex:VISSBindingServiceInterfaceDeploymentExtension
       ">
      <SD GID="VISSMex-version">1.0.0</SD>
      <SDG GID="userDefinedVISSBinding">
        <SDG GID="
           VISSBindingServiceInterfaceDeployment" />
      </SDG>
    </SDG>
  </SDGS>
</ADMIN-DATA>
<FIELD-DEPLOYMENTS>
  <USER-DEFINED-FIELD-DEPLOYMENT>
    <SHORT-NAME>LowVoltageSystemState</SHORT-NAME>
    <FIELD-REF DEST="FIELD">/Catalogs/VSS/
       ServiceInterfaces/Vehicle/Vehicle/
       LowVoltageSystemState</FIELD-REF>
    <GET>
      <SHORT-NAME>getLowVoltageSystemState</SHORT-
         NAME>
      <ADMIN-DATA>
        <SDGS>
          <SDG GID="
             VISSMex:VISSBindingMethodDeploymentExtension
             ">
            <SD GID="VISSMex-version">1.0.0</SD>
            <SDG GID="userDefinedVISSBinding">
              <SDG GID="VISSBindingMethodDeployment"
                 />
            </SDG>
          </SDG>
        </SDGS>
      </ADMIN-DATA>
    </GET>
    <NOTIFIER>
      <SHORT-NAME>onLowVoltageSystemState</SHORT-NAME
         >
      <ADMIN-DATA>
        <SDGS>
          <SDG GID="
             VISSMex:VISSBindingEventDeploymentExtension
             ">
            <SD GID="VISSMex-version">1.0.0</SD>
            <SDG GID="userDefinedVISSBinding">
              <SDG GID="VISSBindingEventDeployment" /
                 >
            </SDG>
          </SDG>
        </SDGS>
      </ADMIN-DATA>
    </NOTIFIER>
  </USER-DEFINED-FIELD-DEPLOYMENT>
</FIELD-DEPLOYMENTS>
```

```
                    </USER-DEFINED-SERVICE-INTERFACE-DEPLOYMENT>
                  </ELEMENTS>
                </AR-PACKAGE>
              </AR-PACKAGES>
            </AR-PACKAGE>
          </AR-PACKAGES>
        </AR-PACKAGE>
      </AR-PACKAGES>
    </AR-PACKAGE>
  </AR-PACKAGES>
</AUTOSAR>
```

Listing A.16: User Defined Service Interface Deployments

A.2.7 Provided Service Instances

```
<?xml version='1.0' encoding='UTF-8'?>
<AUTOSAR xmlns="http://autosar.org/schema/r4.0"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xsi:schemaLocation="http://autosar.org/schema/r4.0 AUTOSAR_00055.xsd
       ">
    <AR-PACKAGES>
        <AR-PACKAGE>
            <SHORT-NAME>APIGateway</SHORT-NAME>
            <ELEMENTS>
                <PROVIDED-USER-DEFINED-SERVICE-INSTANCE>
                    <SHORT-NAME>P_Vehicle</SHORT-NAME>
                    <ADMIN-DATA>
                        <SDGS>
                            <SDG GID="
                               VISSMex:ProvidedVISSServiceInstanceExtension
                               ">
                                <SD GID="VISSMex-version">1.0.0</SD>
                                <SDG GID="userDefinedVISSBinding">
                                    <SDG GID="ProvidedVISSServiceInstance">
                                        <SD GID="vssPath">Vehicle</SD>
                                    </SDG>
                                </SDG>
                            </SDG>
                        </SDGS>
                    </ADMIN-DATA>
                    <!--<SERVICE-INTERFACE-DEPLOYMENT-REF DEST="USER-
                       DEFINED-SERVICE-INTERFACE-DEPLOYMENT">/Catalogs/VSS/
                       VISSServiceInterfaceDeployments/Vehicle</SERVICE-
                       INTERFACE-DEPLOYMENT-REF>-->
                </PROVIDED-USER-DEFINED-SERVICE-INSTANCE>
            </ELEMENTS>

        </AR-PACKAGE>
    </AR-PACKAGES>
</AUTOSAR>
```

Listing A.17: Provided User Defined Service Instance Model

B Change History of AUTOSAR Traceable Items

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

B.1 Traceable item history of this document according to AUTOSAR Release R26-11

B.1.1 Added Specification Items in R26-11

none

B.1.2 Changed Specification Items in R26-11

none

B.1.3 Deleted Specification Items in R26-11

none

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

B.2.1 Added Specification Items in R25-11

none

B.2.2 Changed Specification Items in R25-11

none

B.2.3 Deleted Specification Items in R25-11

none

B.3 Traceable item history of this document according to AUTOSAR Release R24-11

B.3.1 Added Specification Items in R24-11

Number	Heading
[AP_TR_VSS_01010]	Catalog VSS
[AP_TR_VSS_01018]	Package VSS Versions
[AP_TR_VSS_01019]	VSS Branch Mapping for Types as ARPackage Structure
[AP_TR_VSS_02001]	VSS Primitive Implementation Data Types
[AP_TR_VSS_02003]	VSS Primitive String Type
[AP_TR_VSS_02010]	Application Data Type
[AP_TR_VSS_02011]	Application Data Type Name
[AP_TR_VSS_02021]	StdCppImplementationDataType
[AP_TR_VSS_02022]	StdCppImplementationDataType Name
[AP_TR_VSS_02023]	StdCppImplementationDataType Namespace
[AP_TR_VSS_02051]	DataTypeMappingSet Location
[AP_TR_VSS_02052]	DataTypeMapping
[AP_TR_VSS_02061]	Data Constraint per Data Type
[AP_TR_VSS_02062]	Data Constraint Limits
[AP_TR_VSS_02063]	Data Constraint Location
[AP_TR_VSS_02064]	Data Constraint Naming
[AP_TR_VSS_02071]	VSS Primitive Types Boundary Restrictions
[AP_TR_VSS_02072]	VSS String Enum Restrictions
[AP_TR_VSS_02402]	VSS Units
[AP_TR_VSS_03101]	Application Array Data Type
[AP_TR_VSS_03102]	Application Array Data Type Reference
[AP_TR_VSS_03103]	Application Array Size Handling
[AP_TR_VSS_03104]	Application Array Name
[AP_TR_VSS_03105]	Application Array Element Data Type
[AP_TR_VSS_03106]	Application Array Element Name
[AP_TR_VSS_03110]	C++ Implementation Array Data Type
[AP_TR_VSS_03111]	C++ Implementation Array Data Reference
[AP_TR_VSS_03112]	C++ Implementation Array Name
[AP_TR_VSS_03201]	Application Record Data Type
[AP_TR_VSS_03202]	Application Record Name
[AP_TR_VSS_03203]	Application Record Element Type
[AP_TR_VSS_03204]	Application Record Element Type Reference



△

Number	Heading
[AP_TR_VSS_03211]	C++ Implementation Struct Data Type
[AP_TR_VSS_03212]	C++ Implementation Element Type Reference
[AP_TR_VSS_03213]	C++ Implementation Struct Array Name
[AP_TR_VSS_03214]	C++ Implementation Element Array Name
[AP_TR_VSS_04003]	VSS Units Package
[AP_TR_VSS_04401]	Common Unit Mapping
[AP_TR_VSS_04404]	VSS Units Reference
[AP_TR_VSS_04440]	VSS Description Base Structure
[AP_TR_VSS_04441]	VSS Description Base Structure
[AP_TR_VSS_04442]	VSS Description
[AP_TR_VSS_04443]	VSS Comment Base Structure
[AP_TR_VSS_04444]	VSS Comment
[AP_TR_VSS_04450]	VSS Key and Value Base Structure
[AP_TR_VSS_04451]	VSS Key and Value modelling
[AP_TR_VSS_05011]	AutomotiveAPI
[AP_TR_VSS_05201]	Service Interfaces
[AP_TR_VSS_05202]	Data Retrieval
[AP_TR_VSS_05203]	Data Updates
[AP_TR_VSS_05210]	VISS/VSS Service Deployments
[AP_TR_VSS_05211]	VISS/VSS Field Deployments
[AP_TR_VSS_05212]	VISS/VSS Set Method Deployments
[AP_TR_VSS_05213]	VISS/VSS Field Get Deployments
[AP_TR_VSS_05214]	VISS/VSS Field Notifier Deployments
[AP_TR_VSS_05215]	User Defined VISS Service Instance
[AP_TR_VSS_05220]	Provided Service Instance Package Structure
[AP_TR_VSS_05221]	Provided Service Instance
[AP_TR_VSS_05222]	Provided Service Instance VSS Path
[AP_TR_VSS_05223]	Provided Service Instance Deployment Reference
[AP_TR_VSS_05301]	Gateway Software Component
[AP_TR_VSS_05302]	Provided Ports Prototype
[AP_TR_VSS_05303]	Provided Ports Prototype Short Name
[AP_TR_VSS_05304]	Provided Interface
[AP_TR_VSS_05305]	Service Instance to Port Prototype Mapping
[AP_TR_VSS_05306]	Provided Ports Prototype Long Name

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

B.3.2 Changed Specification Items in R24-11

none

B.3.3 Deleted Specification Items in R24-11

none