Document Title	Technical Report on Supplementary Material of Workflow Example
Document Owner	AUTOSAR
Document Responsibility	AUTOSAR
Document Identification No	1145

Document Status	published
Part of AUTOSAR Standard	Classic Platform
Part of Standard Release	R25-11

Document Change History

Date	**Release**	**Changed by**	**Description**
2025-11-27	R25-11	AUTOSAR Release Management	• Initial release

Table of Contents

1 Introduction

This technical report provides additional information to the AUTOSAR Workflow Example and is intended for all those who are concerned with the example.

1.1 Objectives

The objectives for the Workflow Example are:

- AUTOSAR is committed to enhance user experience by providing practical guides and examples to ease the entry into the work with AUTOSAR and to show typical applications.
- Since AUTOSAR specifies exchange artifacts, we want to provide practical examples based on these standards that can be processed by as many implementation providers as possible.
- To verify the usability and quality of these exchange artifacts, the processed artifacts must be integrable and systemically functional after deployment to a hardware target.
- This kind of test corresponds to the idea of a plug-fest, in which different providers plug their products installed on a hardware together and provide proof that they can work together.

1.2 Scope

The basis of the Workflow Example is an iterative process which shall handle the challenges of AUTOSAR projects, e.g.

- specifications still leave room for interpretation,
- incomplete implementations of the standard,
- errors that unfortunately creep-in from time to time.

The workflow example provides a system description from which development partners can generate the system extracts for the individual sub-systems in such a way that as many partners as possible can process them smoothly with their tools.

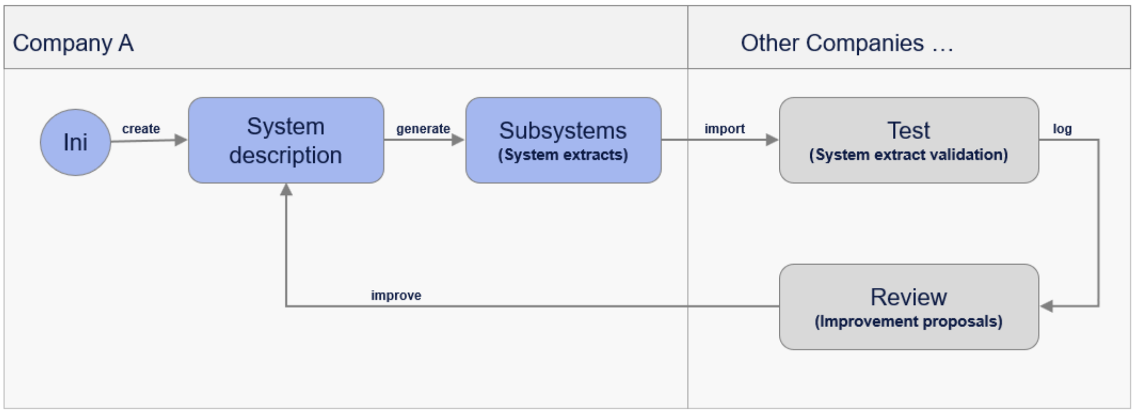


Figure 1.1: Structure of iterative process

2 Example System

The basis function of the system is that a wheel speed which is simulated by a potentiometer, converted by an ADC module (Analog-Digital-Converter) of the WheelSpeed ECU, transmitted via CAN to the VehicleSpeed ECU and the calculated vehicle speed is indicated by two LEDs by a DIO module (Digital-Input-Output).

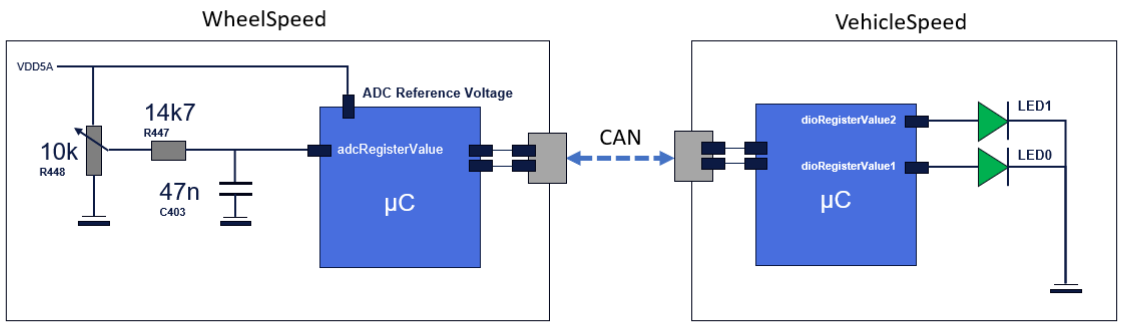
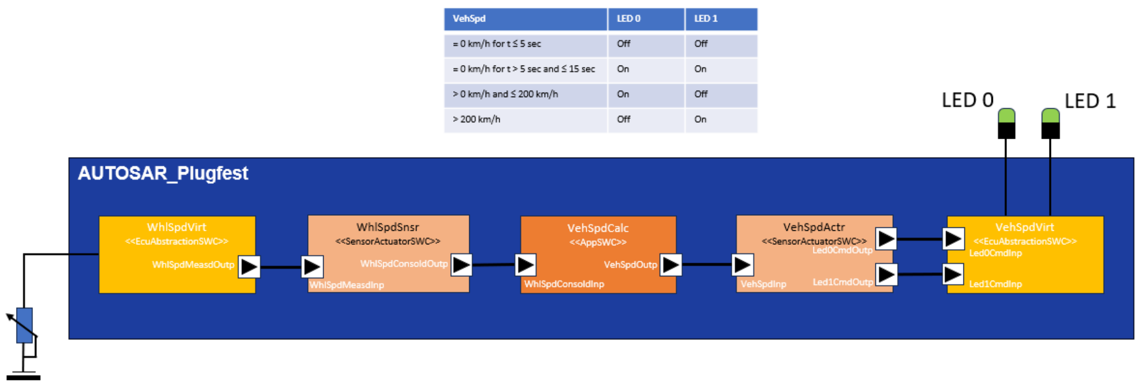


Figure 2.1: Functional sketch of the system

Transferred in the AUTOSAR methodology the function is defined by five software components in one system composition. Two SensorActuatorSwComponentTypes for the interaction with outer world, one SwComponentType for the internal calculation and two EcuAbstractionSwComponentType for the abstraction of the hardware. The System class is used to describe the System Configuration of the complete AUTOSAR system. The category property of System shall indicate the role of this work product. Here the category SYSTEM_DESCRIPTION is used. See [TPS_SYST_01003] in [1]. This is typically done by an OEM.



VehSpd	LED 0	LED 1
= 0 km/h for t ≤ 5 sec	Off	Off
= 0 km/h for t > 5 sec and ≤ 15 sec	On	On
> 0 km/h and ≤ 200 km/h	On	Off
> 200 km/h	Off	On

Figure 2.2: Functional representation in AUTOSAR

For the Workflow Example the overall system composition is separated in two SYSTEM_EXTRACTs assigned each to a separate ECU. These SYSTEM_EXTRACTs (SystemExtract 1 and 2) are provided to two different suppliers.

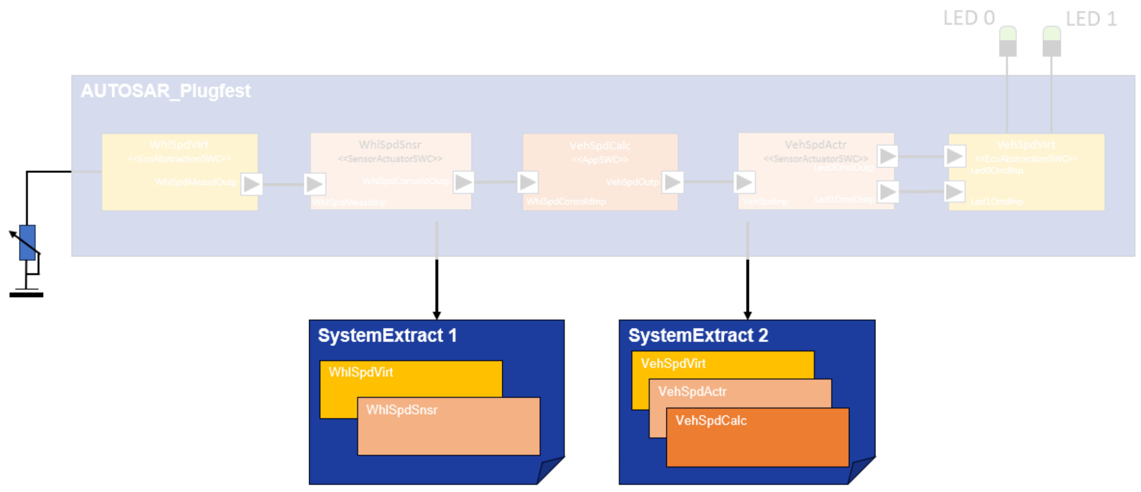


Figure 2.3: Separation in two System Extracts

On the supplier side the `SYSTEM_EXTRACT` is imported and all required data are added to generate a sub-system for an ECU which will be later integrated in the complete system by the original requester.

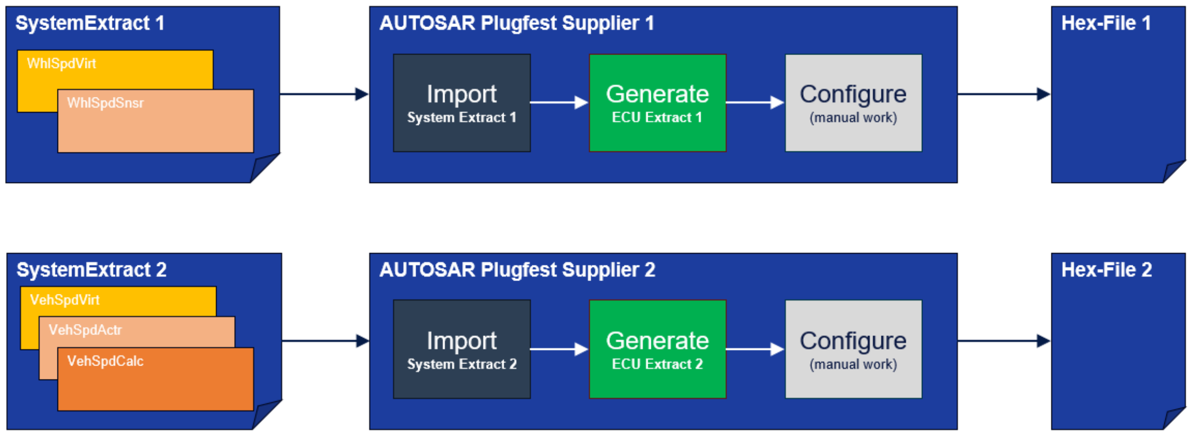


Figure 2.4: Processing of System Extracts on supplier side

The provided `SYSTEM_EXTRACT` contains several methodological aspects defined by the different **T**em**P**late **S**pecifications (TPS documents).

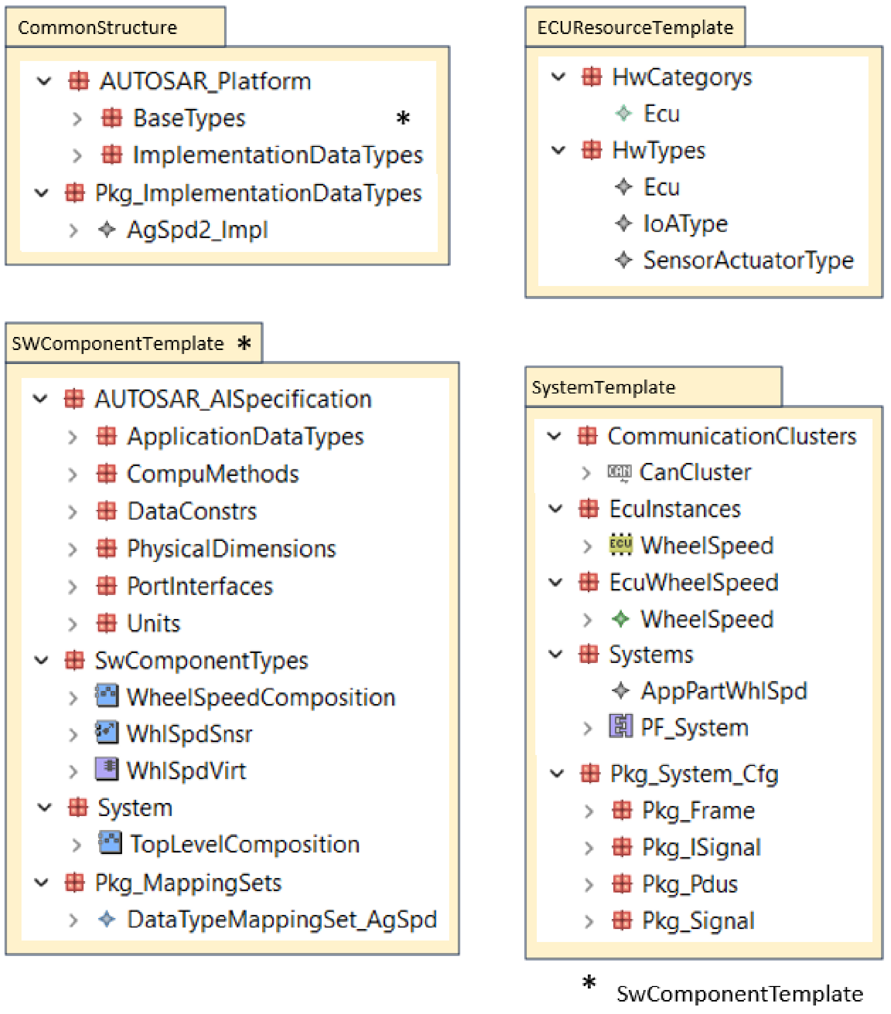


Figure 2.5: Assignment of System Extract artifacts to dedicated Template Specifications

The packaging of the artifacts is described in GenericStructure [2]. A package may contain an arbitrary number of elements, represented by the abstract ARElement. Note that the aggregation of ARElement in ARPackage is subject to variation. Therefore the illustration in 2.5 is only one possible representation. The CommonStructure is described in [2]. The other packages of the meta-model as the SystemTemplate [1], the SWComponentTemplate [3] and the ECUResourceTemplate [4] and their dependencies to each other are described in [2] - Organization of the Meta-Model.

In the last step, the ECUs provided by the different suppliers, will be integrated in the target system.

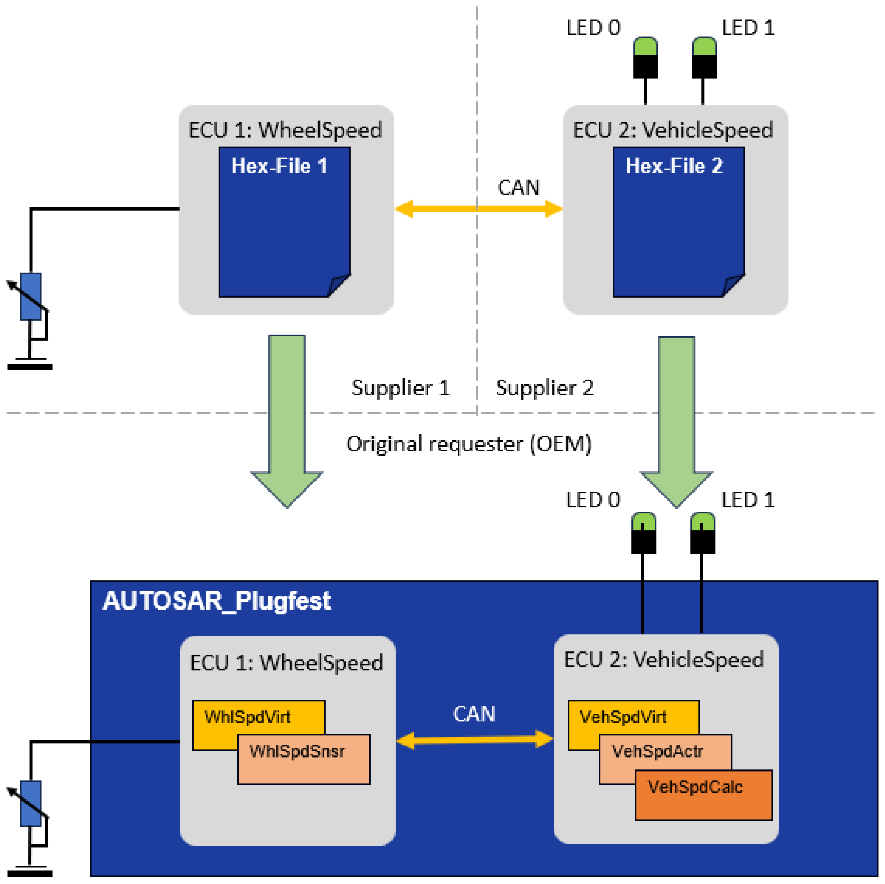


Figure 2.6: Provided ECUs are integrated in target system

2.1 Technical Aspects

The system defined in the `SYSTEM_DESCRIPTION` consists of the "TopLevelComposition" containing two sub-compositions.

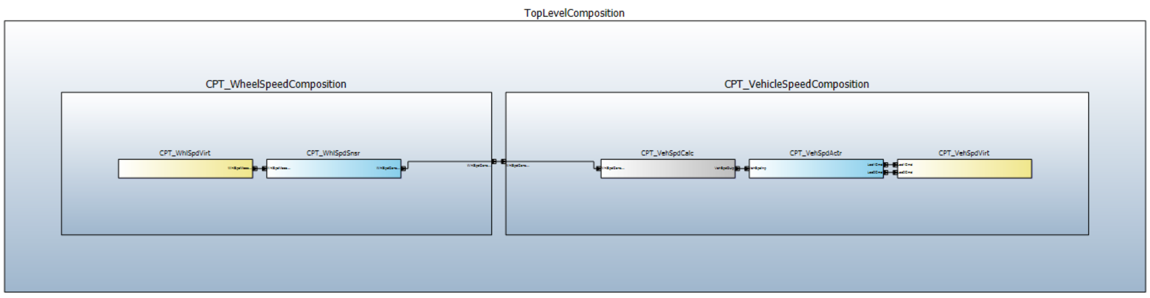


Figure 2.7: TopLevelComposition of designed system

These two sub-compositions are used to generate a `SYSTEM_EXTRACT` for each ECU.

The first `SYSTEM_EXTRACT` is dedicated for the ECU "WheelSpeed".

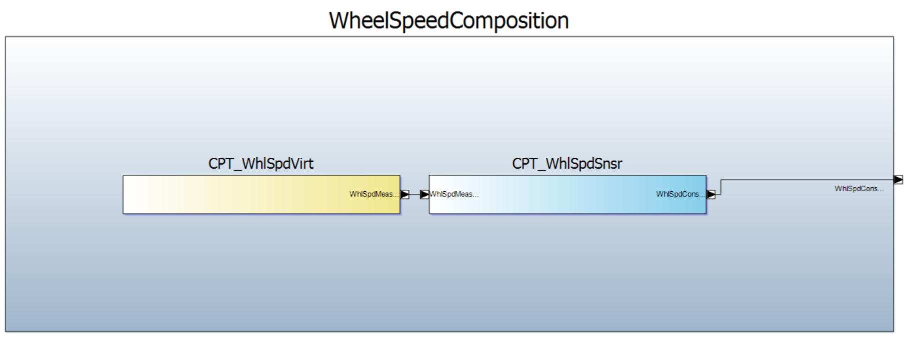


Figure 2.8: TopLevelComposition of sub-system "WheelSpeed"

This composition consists of the EcuAbstractionSwComponentType named "CPT_WhlSpdVirt" which abstracts the hardware and the SensorActuatorSwComponentType named "CPT_WhlSpdSnsr" which handles the sensor signal.

The second SYSTEM_EXTRACT is dedicated for the ECU "VehicleSpeed".

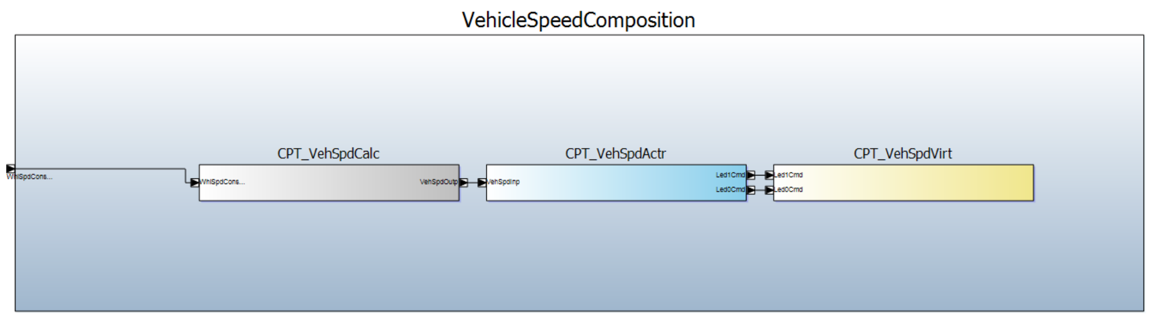


Figure 2.9: TopLevelComposition of sub-system "VehicleSpeed"

This composition consists of three components: the ApplicationSwComponentType "CPT_VehSpdCalc" receives the data from the other ECU and converts the vehicle speed from the wheel speed. The SensorActuatorSwComponentType "CPT_VehSpdActr" prepares the actuator signal for each LED based on the calculated vehicle speed which will be finally received by the EcuAbstractionSwComponentType "CPT_VehSpdVirt" which encasulates the access to the MCAL drivers for the LED.

2.2 Architectural Aspects

Two approaches - top-down and bottom-up - are implemented in this project and will be described in a brief way.

Hint to delivered artefacts!

The first iteration of this project only contains the artefacts of the top-down approach. See appendix A

Top-down Approach

The top-down approach is described in chapter `13.3 SW component inclusion and top level data mapping` [1] and also used in this project.

The system designer of the "TopLevelComposition" can design the system in such a way that

- exchangeability on application side and
- decoupling of data mapping between network architecture and application architecture

are achievable. He creates the "TopLevelComposition" based on the `RootSwCompositionPrototype` and references the `CompositionSwComponentType` as the top-level software composition. The RootSwComposition will usually not contain all atomic software components that are used in a complete VFB (Virtual Function Bus) system.

Instead, the focus is on the interfaces between the components or whole compositions. The system design could even contain only compositions and their ports. In this example, the components are included as well.

The system designer (e.g. OEM) is primarily interested in the required functionality and the interfaces defining the integration of the software compositions or software components into the system. In this project the system designer embeds the two sub-compositions "CPT_WheelSpeedComposition" and "CPT_VehicleSpeedComposition" and the required infrastructure.

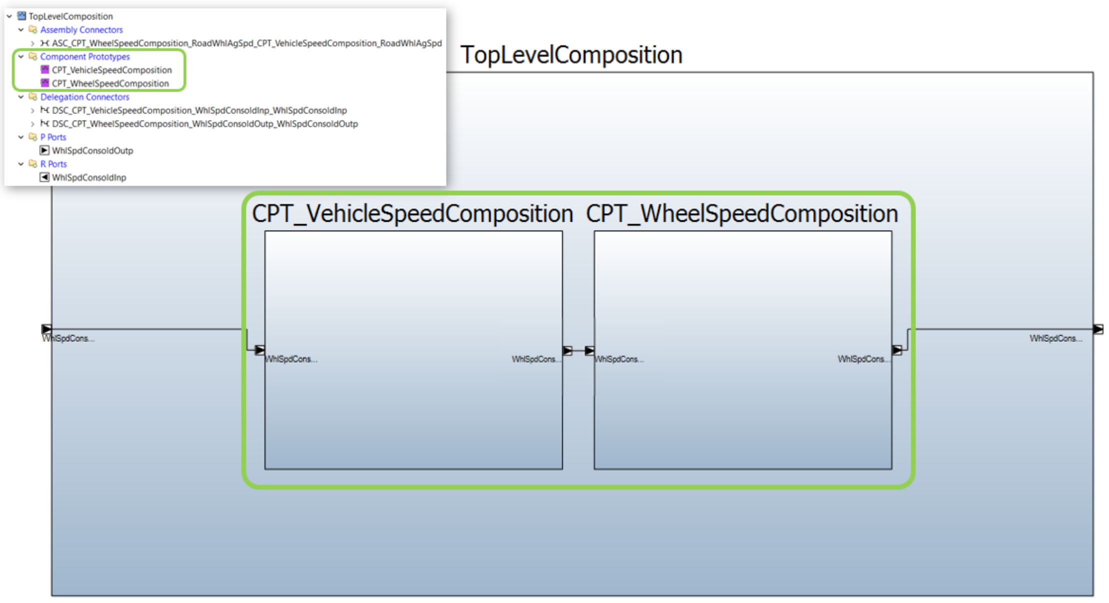
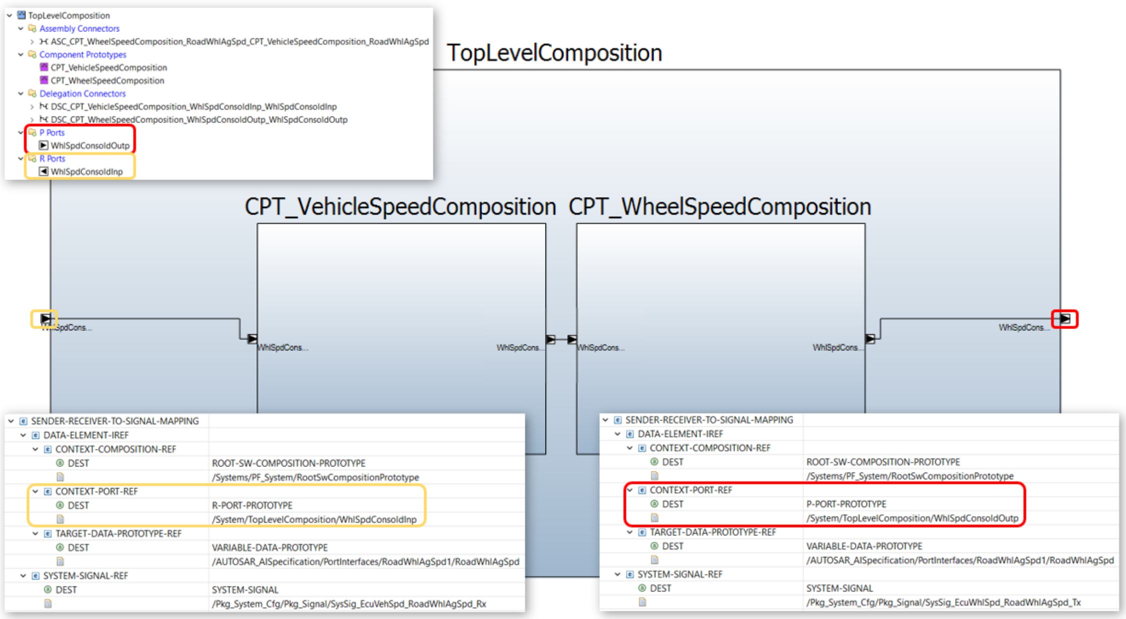

TopLevelComposition
Assembly Connectors
ASC_CPT_WheelSpeedComposition_RoadWhlAgSpd_CPT_VehicleSpeedComposition_RoadWhlAgSpd
Component Prototypes
CPT_VehicleSpeedComposition
CPT_WheelSpeedComposition
Delegation Connectors
DSC_CPT_VehicleSpeedComposition_WhlSpdConsoldInp_WhlSpdConsoldInp
DSC_CPT_WheelSpeedComposition_WhlSpdConsoldOutp_WhlSpdConsoldOutp
P Ports
WhlSpdConsoldOutp
R Ports
WhlSpdConsoldInp
TopLevelComposition
CPT_VehicleSpeedComposition
CPT_WheelSpeedComposition
WhlSpdCons...
WhlSpdCons...
WhlSpdCons...
WhlSpdCons...
WhlSpdCons...
WhlSpdCons...

Figure 2.10: Design of TopLevelComposition with two sub-compositions (top-down approach)

An essential aspect of this architectural approach is to decouple the `DataMapping` of the network communication from that of the implemented applications. The `PortPro-`

totypes realizing the network communication are defined at the "TopLevelComposition". The corresponding DataMapping, implemented by the subclass SenderReceiverToSignalMapping, is also defined referring to the outer PortPrototypes.



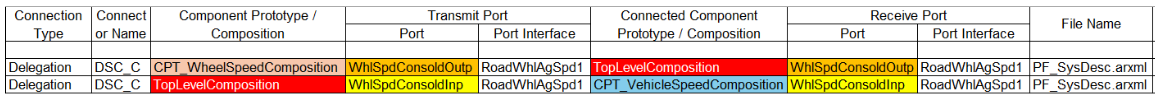
```
TopLevelComposition
  Assembly Connectors
    ASC_CPT_WheelSpeedComposition_RoadWhlAgSpd_CPT_VehicleSpeedComposition_RoadWhlAgSpd
  Component Prototypes
    CPT_VehicleSpeedComposition
    CPT_WheelSpeedComposition
  Delegation Connectors
    DSC_CPT_VehicleSpeedComposition_WhlSpdConsoldInp_WhlSpdConsoldInp
    DSC_CPT_WheelSpeedComposition_WhlSpdConsoldOutp_WhlSpdConsoldOutp
  P Ports
    WhlSpdConsoldOutp
  R Ports
    WhlSpdConsoldInp
```

```
SENDER-RECEIVER-TO-SIGNAL-MAPPING
  DATA-ELEMENT-IREF
    CONTEXT-COMPOSITION-REF
      DEST                  ROOT-SW-COMPOSITION-PROTOTYPE
                            /Systems/PF_System/RootSwCompositionPrototype
    CONTEXT-PORT-REF
      DEST                  R-PORT-PROTOTYPE
                            /System/TopLevelComposition/WhlSpdConsoldInp
    TARGET-DATA-PROTOTYPE-REF
      DEST                  VARIABLE-DATA-PROTOTYPE
                            /AUTOSAR_AISpecification/PortInterfaces/RoadWhlAgSpd1/RoadWhlAgSpd
  SYSTEM-SIGNAL-REF
    DEST                    SYSTEM-SIGNAL
                            /Pkg_System_Cfg/Pkg_Signal/SysSig_EcuVehSpd_RoadWhlAgSpd_Rx
```

```
SENDER-RECEIVER-TO-SIGNAL-MAPPING
  DATA-ELEMENT-IREF
    CONTEXT-COMPOSITION-REF
      DEST                  ROOT-SW-COMPOSITION-PROTOTYPE
                            /Systems/PF_System/RootSwCompositionPrototype
    CONTEXT-PORT-REF
      DEST                  P-PORT-PROTOTYPE
                            /System/TopLevelComposition/WhlSpdConsoldOutp
    TARGET-DATA-PROTOTYPE-REF
      DEST                  VARIABLE-DATA-PROTOTYPE
                            /AUTOSAR_AISpecification/PortInterfaces/RoadWhlAgSpd1/RoadWhlAgSpd
  SYSTEM-SIGNAL-REF
    DEST                    SYSTEM-SIGNAL
                            /Pkg_System_Cfg/Pkg_Signal/SysSig_EcuWhlSpd_RoadWhlAgSpd_Tx
```

Figure 2.11: Outer PortPrototypes realize network communication

From a VFB based point of view the outer PortPrototypes are delegated to the inner PortPrototypes.



Connection Type	Connect or Name	Component Prototype / Composition	Transmit Port		Connected Component Prototype / Composition	Receive Port		File Name
			Port	Port Interface		Port	Port Interface	
Delegation	DSC_C	CPT_WheelSpeedComposition	WhlSpdConsoldOutp	RoadWhlAgSpd1	TopLevelComposition	WhlSpdConsoldOutp	RoadWhlAgSpd1	PF_SysDesc.arxml
Delegation	DSC_C	TopLevelComposition	WhlSpdConsoldInp	RoadWhlAgSpd1	CPT_VehicleSpeedComposition	WhlSpdConsoldInp	RoadWhlAgSpd1	PF_SysDesc.arxml

Figure 2.12: Delegation of outer PortPrototypes to inner ones

The benefit of the described design approach is that actually the software component architecture inside the root composition can be changed without affecting the DataMappings and thus the network definition. Therefore the supplier of the embedded software compositions or software components can act independently.

Bottom-Up Approach

The bottom-up approach is described in chapter 13.2 Data Mapping in the System Extract in [1] and also validated as an alternative design approach in the context of the Workflow Example.

The system design is created bottom-up. In order to assemble applications from AUTOSAR components (AtomicSwComponentType), compositions can be build up hierarchically, until the outermost CompositionSwComponentType forms a kind of top-level composition. This mainly bases on [constr_3031] in [1], which tells that the unique feature of the complete system description is, that it doesn't have any outside ports, but all the software components contained in it are connected to each other (AssemblySwConnector) and fully specified by their SwComponentTypes, PortPrototypes, PortInterfaces, VariableDataPrototypes, InternalBehavior etc..

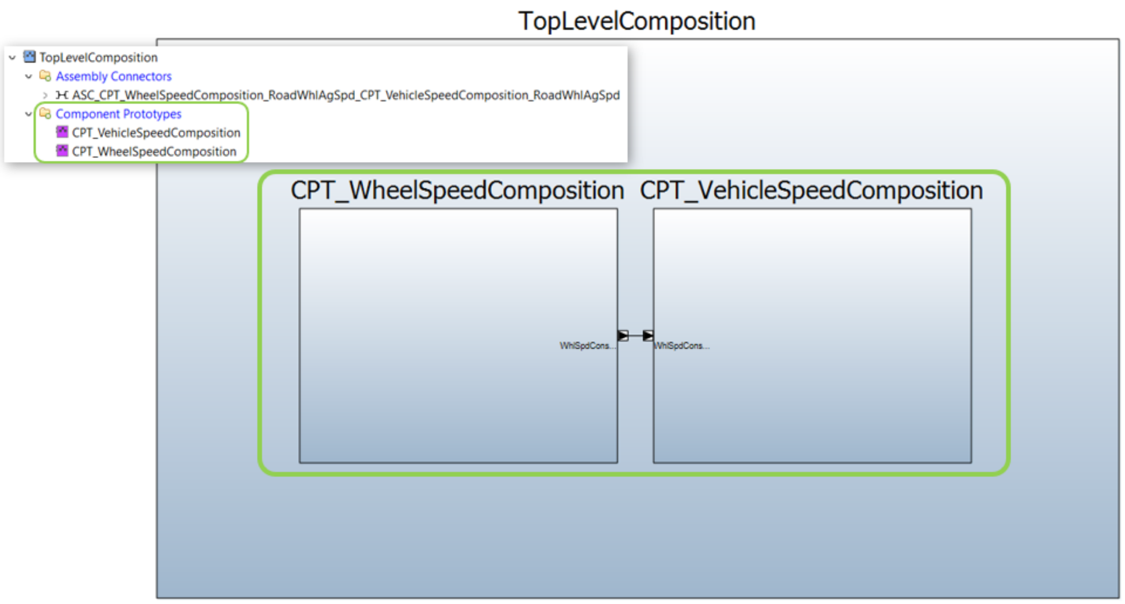
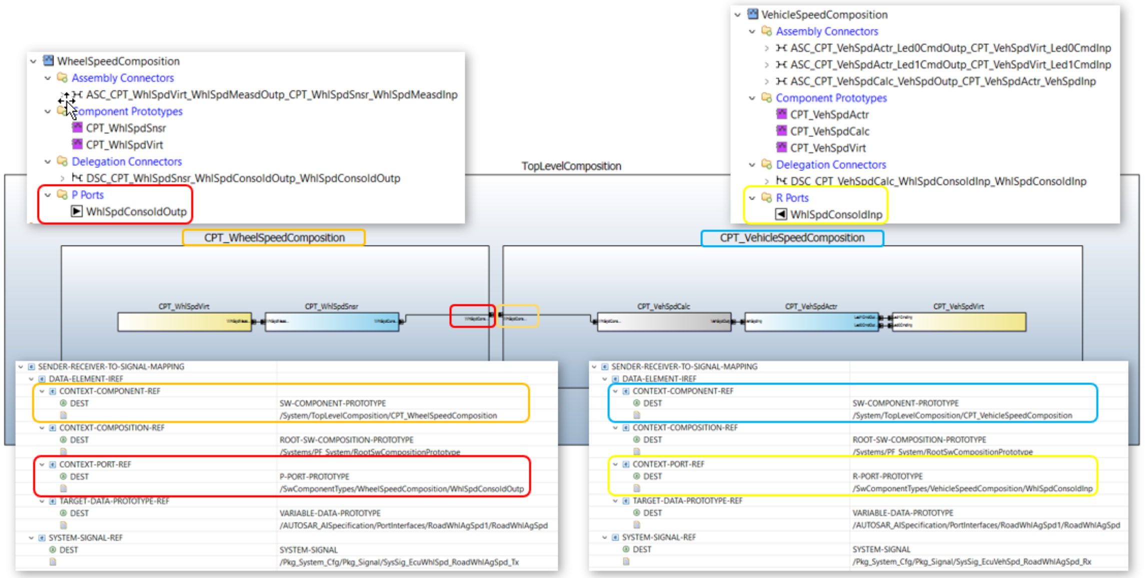

TopLevelComposition
TopLevelComposition
Assembly Connectors
ASC_CPT_WheelSpeedComposition_RoadWhlAgSpd_CPT_VehicleSpeedComposition_RoadWhlAgSpd
Component Prototypes
CPT_VehicleSpeedComposition
CPT_WheelSpeedComposition
CPT_WheelSpeedComposition
CPT_VehicleSpeedComposition
WhlSpdCons...
WhlSpdCons...

Figure 2.13: Design of TopLevelComposition with two sub-compositions (Bottom-Up Approach)

In this approach, it is convenient to attach the DataMapping directly to the software component. The PortPrototypes are assigned to the SwComponentTypes of "CPT_WheelSpeedComposition" and "CPT_VehicleSpeedComposition".

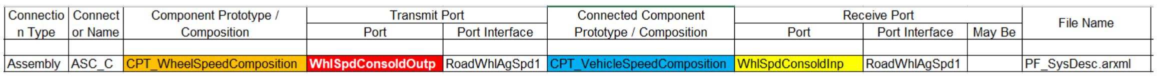

WheelSpeedComposition
Assembly Connectors
ASC_CPT_WhlSpdVirt_WhlSpdMeasdOutp_CPT_WhlSpdSnsr_WhlSpdMeasdInp
Component Prototypes
CPT_WhlSpdSnsr
CPT_WhlSpdVirt
Delegation Connectors
DSC_CPT_WhlSpdSnsr_WhlSpdConsoldOutp_WhlSpdConsoldOutp
P Ports
WhlSpdConsoldOutp
VehicleSpeedComposition
Assembly Connectors
ASC_CPT_VehSpdActr_Led0CmdOutp_CPT_VehSpdVirt_Led0CmdInp
ASC_CPT_VehSpdActr_Led1CmdOutp_CPT_VehSpdVirt_Led1CmdInp
ASC_CPT_VehSpdCalc_VehSpdOutp_CPT_VehSpdActr_VehSpdInp
Component Prototypes
CPT_VehSpdActr
CPT_VehSpdCalc
CPT_VehSpdVirt
Delegation Connectors
DSC_CPT_VehSpdCalc_WhlSpdConsoldInp_WhlSpdConsoldInp
R Ports
WhlSpdConsoldInp
TopLevelComposition
CPT_WheelSpeedComposition
CPT_VehicleSpeedComposition
CPT_WhlSpdVirt
CPT_WhlSpdSnsr
CPT_VehSpdCalc
CPT_VehSpdActr
CPT_VehSpdVirt

```
SENDER-RECEIVER-TO-SIGNAL-MAPPING
  DATA-ELEMENT-IREF
    CONTEXT-COMPONENT-REF
      DEST                         SW-COMPONENT-PROTOTYPE
                                   /System/TopLevelComposition/CPT_WheelSpeedComposition
    CONTEXT-COMPOSITION-REF
      DEST                         ROOT-SW-COMPOSITION-PROTOTYPE
                                   /Systems/PF_System/RootSwCompositionPrototype
    CONTEXT-PORT-REF
      DEST                         P-PORT-PROTOTYPE
                                   /SwComponentTypes/WheelSpeedComposition/WhlSpdConsoldOutp
    TARGET-DATA-PROTOTYPE-REF
      DEST                         VARIABLE-DATA-PROTOTYPE
                                   /AUTOSAR_AISpecification/PortInterfaces/RoadWhlAgSpd1/RoadWhlAgSpd
  SYSTEM-SIGNAL-REF
    DEST                           SYSTEM-SIGNAL
                                   /Pkg_System_Cfg/Pkg_Signal/SysSig_EcuWhlSpd_RoadWhlAgSpd_Tx
```

```
SENDER-RECEIVER-TO-SIGNAL-MAPPING
  DATA-ELEMENT-IREF
    CONTEXT-COMPONENT-REF
      DEST                         SW-COMPONENT-PROTOTYPE
                                   /System/TopLevelComposition/CPT_VehicleSpeedComposition
    CONTEXT-COMPOSITION-REF
      DEST                         ROOT-SW-COMPOSITION-PROTOTYPE
                                   /Systems/PF_System/RootSwCompositionPrototype
    CONTEXT-PORT-REF
      DEST                         R-PORT-PROTOTYPE
                                   /SwComponentTypes/VehicleSpeedComposition/WhlSpdConsoldInp
    TARGET-DATA-PROTOTYPE-REF
      DEST                         VARIABLE-DATA-PROTOTYPE
                                   /AUTOSAR_AISpecification/PortInterfaces/RoadWhlAgSpd1/RoadWhlAgSpd
  SYSTEM-SIGNAL-REF
    DEST                           SYSTEM-SIGNAL
                                   /Pkg_System_Cfg/Pkg_Signal/SysSig_EcuVehSpd_RoadWhlAgSpd_Rx
```

Figure 2.14: PortPrototypes are assigned to SwComponentTypes

Both software compositions are self-contained and connected by AssemblySwConnector on the next higher composition level, here the "TopLevelComposition", see figure 2.13.



Connection Type	Connector Name	Component Prototype / Composition	Transmit Port		Connected Component Prototype / Composition	Receive Port			File Name
			Port	Port Interface		Port	Port Interface	May Be	
Assembly	ASC_C	CPT_WheelSpeedComposition	**WhlSpdConsoldOutp**	RoadWhlAgSpd1	CPT_VehicleSpeedComposition	WhlSpdConsoldInp	RoadWhlAgSpd1		PF_SysDesc.arxml

Figure 2.15: Connection of software compositions by AssemblySwConnector on next higher composition level

The benefit of the described design approach is that the designer of an enveloping composition uses the functionalities to be included, in form of software components or compositions, and builds up stepwise the complete system description. This is helpful if the OEM is not able to provide an system description at all.

3 Related Documentation

[1] System Template
AUTOSAR_CP_TPS_SystemTemplate

[2] Generic Structure Template
AUTOSAR_FO_TPS_GenericStructureTemplate

[3] Software Component Template
AUTOSAR_CP_TPS_SoftwareComponentTemplate

[4] Specification of ECU Resource Template
AUTOSAR_CP_TPS_ECUResourceTemplate

[5] AUTOSAR Webpage – Classic Platform
https://www.autosar.org/standards/classic-platform

A Delivered artefacts

The delivery of the first iteration of the Workflow Example (*WorkflowExample.zip*) only includes the artefacts of the top-down approach, see section 2.2:

These are:

- `ECU_SYSTEM_DESCRIPTION` of the Workflow Example project containing all relevant ARXML files regarding to the infrastructure, software components and system configuration. Used and provided by the OEM.
- `SYSTEM_EXTRACT` of ECU 1 for the WheelSpeed composition (*WheelSpeed__SystemExtract.arxml*), used by the supplier of ECU 1 for further processing.
- `SYSTEM_EXTRACT` of ECU 2 for the VehicleSpeed composition (*VehicleSpeed__SystemExtract.arxml*), used by the supplier of ECU 2 for further processing.
- *AUTOSAR_CP_TR_WorkflowExample.pdf*, documentation of the Workflow Example, this file.

Instruction for user of delivered artefacts!

1. Download the *WorkflowExample.zip* from the AUTOSAR Webpage - Classic Platform [5].
2. Unzip *WorkflowExample.zip*, three folders will be created:
   - EcuSystemDescription,
   - Ecu1SystemExtract and
   - Ecu2SystemExtract.
3. Depending on your role (OEM, supplier) load the corresponding folder in your AUTOSAR tooling and start working. As an `OEM` work based on the EcuSystemDescription, as a `supplier` use the System Extracts of ECU1 or ECU2.
4. As `supplier` continue as illustrated in figure 2.4 and complete the sub-system.
5. As `OEM` integrate the provided ECUs as illustrated in figure 2.6.
6. Job done.

B Mentioned Class Tables

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

Class	***ARElement*** (abstract)			
Note	An element that can be defined stand-alone, i.e. without being part of another element (except for packages of course).			
Base	*ARObject*, *CollectableElement*, *Identifiable*, *MultilanguageReferrable*, *PackageableElement*, *Referrable*			
Subclasses	AclObjectSet, AclOperation, AclPermission, AclRole, AliasNameSet, ApplicabilityInfoSet, Application Partition, *AutosarDataType*, *BaseType*, BlueprintMappingSet, BswEntryRelationshipSet, BswModule Description, BswModuleEntry, BuildActionManifest, CalibrationParameterValueSet, ClientIdDefinitionSet, ClientServerInterfaceToBswModuleEntryBlueprintMapping, Collection, CompuMethod, Consistency NeedsBlueprintSet, ConstantSpecification, ConstantSpecificationMappingSet, CpSoftwareCluster, Cp SoftwareClusterBinaryManifestDescriptor, CpSoftwareClusterMappingSet, CpSoftwareClusterResource Pool, CryptoEllipticCurveProps, CryptoServiceCertificate, CryptoServiceKey, CryptoServicePrimitive, CryptoServiceQueue, CryptoSignatureScheme, DataConstr, DataTransformationSet, DataTypeMapping Set, DdsCpConfig, *DiagnosticCommonElement*, DiagnosticConnection, DiagnosticContributionSet, Dlt ArgumentPropsSet, DltContext, DltEcu, Documentation, E2EProfileCompatibilityProps, EcucDefinition Collection, EcucDestinationUriDefSet, EcucModuleConfigurationValues, EcucModuleDef, EcucValue Collection, EthIpProps, EthTcpIpIcmpProps, EthTcpIpProps, EvaluatedVariantSet, FMFeature, FMFeatureMap, FMFeatureModel, FMFeatureSelectionSet, FirewallRule, FlatMap, GeneralPurpose Connection, HwCategory, HwElement, HwType, *IEEE1722TpConnection*, IPSecConfigProps, IPv6Ext HeaderFilterSet, *IdsCommonElement*, IdsDesign, *Implementation*, ImpositionTimeDefinitionGroup, InterpolationRoutineMappingSet, J1939ControllerApplication, KeywordSet, LifeCycleInfoSet, LifeCycle StateDefinitionGroup, LogAndTraceMessageCollectionSet, MacSecGlobalKayProps, MacSecParticipant Set, McFunction, McGroup, ModeDeclarationGroup, ModeDeclarationMappingSet, OsTaskProxy, PhysicalDimension, PhysicalDimensionMappingSet, *PortInterface*, PortInterfaceMappingSet, Port PrototypeBlueprint, PostBuildVariantCriterion, PostBuildVariantCriterionValueSet, PredefinedVariant, RapidPrototypingScenario, SdgDef, *SecureComProps*, SignalServiceTranslationPropsSet, SomeipSd ClientEventGroupTimingConfig, SomeipSdClientServiceInstanceConfig, SomeipSdServerEventGroup TimingConfig, SomeipSdServerServiceInstanceConfig, SwAddrMethod, SwAxisType, SwComponent MappingConstraints, *SwComponentType*, SwRecordLayout, SwSystemconst, SwSystemconstantValue Set, SwcBswMapping, System, SystemComSpecDefinitionSet, SystemSignal, SystemSignalGroup, TDCpSoftwareClusterMappingSet, TcpOptionFilterSet, *TimingExtension*, TlsConnectionGroup, TlvData IdDefinitionSet, TransformationPropsSet, Unit, UnitGroup, *UploadablePackageElement*, ViewMapSet			
Aggregated by	ARPackage.element			
Attribute	***Type***	***Mult.***	***Kind***	***Note***
–	–	–	–	–

Table B.1: ARElement

Class	**ARPackage**			
Note	AUTOSAR package, allowing to create top level packages to structure the contained ARElements. 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*, *AtpBlueprintable*, *CollectableElement*, *Identifiable*, *MultilanguageReferrable*, *Referrable*			
Aggregated by	ARPackage.arPackage, AUTOSAR.arPackage			
Attribute	***Type***	***Mult.***	***Kind***	***Note***

▽

△

Class	**ARPackage**			

arPackage	ARPackage	*	aggr	This represents a sub package within an ARPackage, thus allowing for an unlimited package hierarchy.
				Stereotypes: atpSplitable; atpVariation
				Tags:
				atp.Splitkey=arPackage.shortName, arPackage.variation Point.shortLabel
				vh.latestBindingTime=blueprintDerivationTime
				xml.sequenceOffset=30

element	PackageableElement	*	aggr	Elements that are part of this package
				Stereotypes: atpSplitable; atpVariation
				Tags:
				atp.Splitkey=element.shortName, element.variation Point.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. **Stereotypes:** atpSplitable **Tags:** atp.Splitkey=referenceBase.shortLabel xml.sequenceOffset=10

Table B.2: ARPackage

Class	**ApplicationSwComponentType**			
Note	The `ApplicationSwComponentType` is used to represent the application software. **Tags:** atp.recommendedPackage=SwComponentTypes			
Base	*ARElement*, *ARObject*, *AtomicSwComponentType*, *AtpBlueprint*, *AtpBlueprintable*, *AtpClassifier*, *Atp Type*, *CollectableElement*, *Identifiable*, *MultilanguageReferrable*, *PackageableElement*, *Referrable*, *Sw ComponentType*			
Aggregated by	ARPackage.element			
Attribute	***Type***	***Mult.***	***Kind***	***Note***
–	–	–	–	–

Table B.3: ApplicationSwComponentType

Class	**AssemblySwConnector**			
Note	`AssemblySwConnector`s are exclusively used to connect `SwComponentPrototype`s in the context of a `CompositionSwComponentType`.			
Base	*ARObject*, *AtpClassifier*, *AtpFeature*, *AtpStructureElement*, *Identifiable*, *MultilanguageReferrable*, *Referrable*, *SwConnector*			
Aggregated by	*AtpClassifier*.atpFeature, CompositionSwComponentType.connector			
Attribute	***Type***	***Mult.***	***Kind***	***Note***
provider	AbstractProvidedPort Prototype	0..1	iref	Instance of providing port. **InstanceRef implemented by:** PPortInComposition InstanceRef
requester	AbstractRequiredPort Prototype	0..1	iref	Instance of requiring port. **InstanceRef implemented by:** RPortInComposition InstanceRef

Table B.4: AssemblySwConnector

Class	***AtomicSwComponentType*** (abstract)			
Note	An atomic software component is atomic in the sense that it cannot be further decomposed and distributed across multiple ECUs.			
Base	*ARElement*, *ARObject*, *AtpBlueprint*, *AtpBlueprintable*, *AtpClassifier*, *AtpType*, *CollectableElement*, *Identifiable*, *MultilanguageReferrable*, *PackageableElement*, *Referrable*, *SwComponentType*			
Subclasses	ApplicationSwComponentType, ComplexDeviceDriverSwComponentType, EcuAbstractionSwComponentType, NvBlockSwComponentType, SensorActuatorSwComponentType, ServiceProxySwComponentType, ServiceSwComponentType			
Aggregated by	ARPackage.element			
Attribute	**Type**	**Mult.**	**Kind**	**Note**
internalBehavior	SwcInternalBehavior	0..1	aggr	The `SwcInternalBehavior`s owned by an `AtomicSwComponentType` can be located in a different physical file. Therefore the aggregation is <<atpSplitable>>. **Stereotypes:** atpSplitable; atpVariation **Tags:** atp.Splitkey=internalBehavior.shortName, internalBehavior.variationPoint.shortLabel vh.latestBindingTime=preCompileTime
symbolProps	SymbolProps	0..1	aggr	This represents the `SymbolProps` for the `AtomicSwComponentType`. **Stereotypes:** atpSplitable **Tags:** atp.Splitkey=symbolProps.shortName

Table B.5: AtomicSwComponentType

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*, *AtpBlueprintable*, *AtpClassifier*, *AtpType*, *CollectableElement*, *Identifiable*, *MultilanguageReferrable*, *PackageableElement*, *Referrable*, *SwComponentType*			
Aggregated by	ARPackage.element			
Attribute	**Type**	**Mult.**	**Kind**	**Note**
component	SwComponentPrototype	*	aggr	The instantiated components that are part of this composition. The aggregation of `SwComponentPrototype` is subject to variability with the purpose to support the conditional existence of a `SwComponentPrototype`. Please be aware: if the conditional existence of `SwComponentPrototype`s is resolved post-build, the deselected `SwComponentPrototype`s are still contained in the ECUs build but the instances are inactive in that they are not scheduled by the RTE. The aggregation is marked as atpSplitable in order to allow the addition of service components to the ECU extract during the ECU integration. The use case for having 0 components owned by the `CompositionSwComponentType` could be to deliver an empty `CompositionSwComponentType` to e.g. a supplier for filling the internal structure. **Stereotypes:** atpSplitable; atpVariation **Tags:** atp.Splitkey=component.shortName, component.variationPoint.shortLabel vh.latestBindingTime=postBuild

▽

△

Class	**CompositionSwComponentType**			
connector	SwConnector	*	aggr	SwConnectors have the principal ability to establish a connection among PortPrototypes. They can have many roles in the context of a CompositionSwComponentType. Details are refined by subclasses. The aggregation of SwConnectors is subject to variability with the purpose to support variant data flow. The aggregation is marked as atpSplitable in order to allow the extension of the ECU extract with AssemblySwConnectors between ApplicationSwComponentTypes and ServiceSwComponentTypes during the ECU integration. **Stereotypes:** atpSplitable; atpVariation **Tags:** atp.Splitkey=connector.shortName, connector.variation Point.shortLabel vh.latestBindingTime=postBuild
constantValue Mapping	ConstantSpecification MappingSet	*	ref	Reference to the ConstantSpecificationMapping to be applied for initValues of PPortComSpecs and RPortComSpec. **Stereotypes:** atpSplitable **Tags:** atp.Splitkey=constantValueMapping
dataType Mapping	DataTypeMappingSet	*	ref	Reference to the DataTypeMappingSet to be applied for the used ApplicationDataTypes in PortInterfaces. Background: when developing subsystems it may happen that ApplicationDataTypes are used on the surface of CompositionSwComponentTypes. In this case it would be reasonable to be able to also provide the intended mapping to the ImplementationDataTypes. However, this mapping shall be informal and not technically binding for the implementors mainly because the RTE generator is not concerned about the CompositionSwComponentTypes. Rationale: if the mapping of ApplicationDataTypes on the delegated and inner PortPrototype matches then the mapping to ImplementationDataTypes is not impacting compatibility. **Stereotypes:** atpSplitable **Tags:** atp.Splitkey=dataTypeMapping
instantiation RTEEventProps	InstantiationRTEEvent Props	*	aggr	This allows to define instantiation specific properties for RTE Events, in particular for instance specific scheduling. **Stereotypes:** atpSplitable; atpVariation **Tags:** atp.Splitkey=instantiationRTEEventProps.shortLabel, instantiationRTEEventProps.variationPoint.shortLabel vh.latestBindingTime=codeGenerationTime This Attribute is only used by the AUTOSAR Classic Platform.
physical Dimension Mapping	PhysicalDimension MappingSet	0..1	ref	This reference identifies the PhysicalDimensionMappingSet that is applicable in the context of the enclosing CompositionSwComponentType. The PhysicalDimensionMappings contained in the PhysicalDimensionMappingSet shall be taken into account for the assessment of the compatibility of PhysicalDimensions in the context of creation of a PortInterfaceMapping in the scope of the CompositionSwComponentType.

Table B.6: CompositionSwComponentType

Class	***DataMapping*** (abstract)			
Note	Mapping of port elements (data elements and parameters) to frames and signals.			
Base	*ARObject*			
Subclasses	ClientServerToSignalMapping, SenderReceiverCompositeElementToSignalMapping, SenderReceiverToSignalGroupMapping, SenderReceiverToSignalMapping, TriggerToSignalMapping			
Aggregated by	SystemMapping.dataMapping			
Attribute	***Type***	***Mult.***	***Kind***	***Note***
introduction	DocumentationBlock	0..1	aggr	This represents introductory documentation about the data mapping.

Table B.7: DataMapping

Class	**EcuAbstractionSwComponentType**			
Note	The ECUAbstraction is a special AtomicSwComponentType that resides between a software-component that wants to access ECU periphery and the Microcontroller Abstraction. The EcuAbstractionSwComponentType introduces the possibility to link from the software representation to its hardware description provided by the ECU Resource Template. **Tags:** atp.recommendedPackage=SwComponentTypes			
Base	*ARElement*, *ARObject*, *AtomicSwComponentType*, *AtpBlueprint*, *AtpBlueprintable*, *AtpClassifier*, *AtpType*, *CollectableElement*, *Identifiable*, *MultilanguageReferrable*, *PackageableElement*, *Referrable*, *SwComponentType*			
Aggregated by	ARPackage.element			
Attribute	***Type***	***Mult.***	***Kind***	***Note***
hardwareElement	HwDescriptionEntity	*	ref	Reference from the EcuAbstractionComponentType to the description of the used HwElements.

Table B.8: EcuAbstractionSwComponentType

Class	***InternalBehavior*** (abstract)			
Note	Common base class (abstract) for the internal behavior of both software components and basic software modules/clusters.			
Base	*ARObject*, *AtpClassifier*, *AtpFeature*, *AtpStructureElement*, *Identifiable*, *MultilanguageReferrable*, *Referrable*			
Subclasses	BswInternalBehavior, SwcInternalBehavior			
Aggregated by	*AtpClassifier*.atpFeature			
Attribute	***Type***	***Mult.***	***Kind***	***Note***
constantMemory	ParameterDataPrototype	*	aggr	Describes a read only memory object containing characteristic value(s) implemented by this InternalBehavior. The shortName of ParameterDataPrototype has to be equal to the "C' identifier of the described constant. The characteristic value(s) might be shared between SwComponentPrototypes of the same SwComponentType. The aggregation of constantMemory is subject to variability with the purpose to support variability in the software component or module implementations. Typically different algorithms in the implementation are requiring different number of memory objects. **Stereotypes:** atpSplitable; atpVariation **Tags:** atp.Splitkey=constantMemory.shortName, constantMemory.variationPoint.shortLabel vh.latestBindingTime=preCompileTime

▽

△

Class	***InternalBehavior*** (abstract)			
constantValue Mapping	ConstantSpecification MappingSet	*	ref	Reference to the ConstantSpecificationMapping to be applied for the particular InternalBehavior **Stereotypes:** atpSplitable **Tags:** atp.Splitkey=constantValueMapping
dataType Mapping	DataTypeMappingSet	*	ref	Reference to the DataTypeMapping to be applied for the particular InternalBehavior **Stereotypes:** atpSplitable **Tags:** atp.Splitkey=dataTypeMapping
exclusiveArea	ExclusiveArea	*	aggr	This specifies an ExclusiveArea for this InternalBehavior. The exclusiveArea is local to the component resp. module. The aggregation of ExclusiveAreas is subject to variability. Note: the number of ExclusiveAreas might vary due to the conditional existence of RunnableEntities or BswModuleEntities. **Stereotypes:** atpSplitable; atpVariation **Tags:** atp.Splitkey=exclusiveArea.shortName, exclusive Area.variationPoint.shortLabel vh.latestBindingTime=preCompileTime
exclusiveArea NestingOrder	ExclusiveAreaNesting Order	*	aggr	This represents the set of ExclusiveAreaNestingOrder owned by the InternalBehavior. **Stereotypes:** atpSplitable; atpVariation **Tags:** atp.Splitkey=exclusiveAreaNestingOrder.shortName, exclusiveAreaNestingOrder.variationPoint.shortLabel vh.latestBindingTime=preCompileTime
staticMemory	VariableDataPrototype	*	aggr	Describes a read and writeable static memory object representing measurerment variables implemented by this software component. The term "static" is used in the meaning of "non-temporary" and does not necessarily specify a linker encapsulation. This kind of memory is only supported if supportsMultipleInstantiation is FALSE. The shortName of the VariableDataPrototype has to be equal with the ”C’ identifier of the described variable. The aggregation of staticMemory is subject to variability with the purpose to support variability in the software component’s implementations. Typically different algorithms in the implementation are requiring different number of memory objects. **Stereotypes:** atpSplitable; atpVariation **Tags:** atp.Splitkey=staticMemory.shortName, static Memory.variationPoint.shortLabel vh.latestBindingTime=preCompileTime

Table B.9: InternalBehavior

Class	***PortInterface*** (abstract)			
Note	Abstract base class for an interface that is either provided or required by a port of a software component.			
Base	*ARElement*, *ARObject*, *AtpBlueprint*, *AtpBlueprintable*, *AtpClassifier*, *AtpType*, *CollectableElement*, *Identifiable*, *MultilanguageReferrable*, *PackageableElement*, *Referrable*			
Subclasses	ClientServerInterface, *DataInterface*, ModeSwitchInterface, TriggerInterface			
Aggregated by	ARPackage.element			
Attribute	***Type***	***Mult.***	***Kind***	***Note***

▽

△

Class	***PortInterface*** (abstract)			
isService	Boolean	0..1	attr	This flag is set if the `PortInterface` is to be used for communication between an • `ApplicationSwComponentType` or • `ServiceProxySwComponentType` or • `SensorActuatorSwComponentType` or • `ComplexDeviceDriverSwComponentType` • `ServiceSwComponentType` • `EcuAbstractionSwComponentType` and a `ServiceSwComponentType` (namely an AUTOSAR Service) located on the same ECU. Otherwise the flag is not set. **Stereotypes:** atpVariation **Tags:** vh.latestBindingTime=blueprintDerivationTime This Attribute is only used by the AUTOSAR Classic Platform.
serviceKind	ServiceProviderEnum	0..1	attr	This attribute provides further details about the nature of the applied service. This Attribute is only used by the AUTOSAR Classic Platform.

Table B.10: PortInterface

Class	***PortPrototype*** (abstract)			
Note	Base class for the ports of an AUTOSAR software component. The aggregation of PortPrototypes is subject to variability with the purpose to support the conditional existence of ports.			
Base	*ARObject*, *AtpBlueprintable*, *AtpFeature*, *AtpPrototype*, *Identifiable*, *MultilanguageReferrable*, *Referrable*			
Subclasses	*AbstractProvidedPortPrototype*, *AbstractRequiredPortPrototype*			
Aggregated by	*AtpClassifier*.atpFeature, *SwComponentType*.port			
Attribute	***Type***	***Mult.***	***Kind***	***Note***
clientServerAnnotation	ClientServerAnnotation	*	aggr	Annotation of this `PortPrototype` with respect to client/server communication.
delegatedPortAnnotation	DelegatedPortAnnotation	0..1	aggr	Annotations on this delegated port.
ioHwAbstractionServerAnnotation	IoHwAbstractionServerAnnotation	*	aggr	Annotations on this IO Hardware Abstraction port.
modePortAnnotation	ModePortAnnotation	*	aggr	Annotations on this mode port.
nvDataPortAnnotation	NvDataPortAnnotation	*	aggr	Annotations on this non voilatile data port.
parameterPortAnnotation	ParameterPortAnnotation	*	aggr	Annotations on this parameter port.
senderReceiverAnnotation	SenderReceiverAnnotation	*	aggr	Collection of annotations of this ports sender/receiver communication. **Stereotypes:** atpSplitable **Tags:** atp.Splitkey=senderReceiverAnnotation
triggerPortAnnotation	TriggerPortAnnotation	*	aggr	Annotations on this trigger port.

Table B.11: PortPrototype

Class	**RootSwCompositionPrototype**			
Note	The RootSwCompositionPrototype represents the top-level-composition of software components within a given System. According to the use case of the System, this may for example be a more or less complete VFB description, the software of a System Extract or the software of a flat ECU Extract with only atomic SWCs. Therefore the RootSwComposition will only occasionally contain all atomic software components that are used in a complete VFB System. The OEM is primarily interested in the required functionality and the interfaces defining the integration of the Software Component into the System. The internal structure of such a component contains often substantial intellectual property of a supplier. Therefore a top-level software composition will often contain empty compositions which represent subsystems. The contained SwComponentPrototypes are fully specified by their SwComponentTypes (including Port Prototypes, PortInterfaces, VariableDataPrototypes, SwcInternalBehavior etc.), and their ports are interconnected using SwConnectorPrototypes.			
Base	*ARObject*, *AtpFeature*, *AtpPrototype*, *Identifiable*, *MultilanguageReferrable*, *Referrable*			
Aggregated by	*AtpClassifier*.atpFeature, System.rootSoftwareComposition			
Attribute	***Type***	***Mult.***	***Kind***	***Note***
calibration ParameterValue Set	CalibrationParameter ValueSet	*	ref	Used CalibrationParameterValueSet for instance specific initialization of calibration parameters. **Stereotypes:** atpSplitable **Tags:** atp.Splitkey=calibrationParameterValueSet This Attribute is only used by the AUTOSAR Classic Platform.
flatMap	FlatMap	0..1	ref	The FlatMap used in the scope of this RootSw CompositionPrototype. **Stereotypes:** atpSplitable **Tags:** atp.Splitkey=flatMap This Attribute is only used by the AUTOSAR Classic Platform.
software Composition	CompositionSw ComponentType	0..1	tref	We assume that there is exactly one top-level composition that includes all Component instances of the system. **Stereotypes:** isOfType

Table B.12: RootSwCompositionPrototype

Class	**SenderReceiverToSignalMapping**			
Note	Mapping of a sender receiver communication data element to a signal.			
Base	*ARObject*, *DataMapping*			
Aggregated by	SystemMapping.dataMapping			
Attribute	***Type***	***Mult.***	***Kind***	***Note***
dataElement	VariableDataPrototype	0..1	iref	Reference to the data element. **InstanceRef implemented by:** VariableDataPrototypeIn SystemInstanceRef
senderToSignal TextTable Mapping	TextTableMapping	0..1	aggr	This mapping allows for the text-table translation between the sending DataPrototype that is defined in the Port Prototype and the physicalProps defined for the System Signal.
signalTo ReceiverText TableMapping	TextTableMapping	0..1	aggr	This mapping allows for the text-table translation between the physicalProps defined for the SystemSignal and a receiving DataPrototype that is defined in the Port Prototype.
systemSignal	SystemSignal	0..1	ref	Reference to the system signal used to carry the data element.

Table B.13: SenderReceiverToSignalMapping

Class	**SensorActuatorSwComponentType**			
Note	The SensorActuatorSwComponentType introduces the possibility to link from the software representation of a sensor/actuator to its hardware description provided by the ECU Resource Template. **Tags:** atp.recommendedPackage=SwComponentTypes			
Base	*ARElement*, *ARObject*, *AtomicSwComponentType*, *AtpBlueprint*, *AtpBlueprintable*, *AtpClassifier*, *AtpType*, *CollectableElement*, *Identifiable*, *MultilanguageReferrable*, *PackageableElement*, *Referrable*, *SwComponentType*			
Aggregated by	ARPackage.element			
Attribute	***Type***	***Mult.***	***Kind***	***Note***
sensorActuator	HwDescriptionEntity	0..1	ref	Reference from the Sensor Actuator Software Component Type to the description of the actual hardware.

Table B.14: SensorActuatorSwComponentType

Class	***SwComponentType*** (abstract)			
Note	Base class for AUTOSAR software components.			
Base	*ARElement*, *ARObject*, *AtpBlueprint*, *AtpBlueprintable*, *AtpClassifier*, *AtpType*, *CollectableElement*, *Identifiable*, *MultilanguageReferrable*, *PackageableElement*, *Referrable*			
Subclasses	*AtomicSwComponentType*, CompositionSwComponentType, ParameterSwComponentType			
Aggregated by	ARPackage.element			
Attribute	***Type***	***Mult.***	***Kind***	***Note***
consistencyNeeds	ConsistencyNeeds	*	aggr	This represents the collection of `ConsistencyNeeds` owned by the enclosing `SwComponentType`. **Stereotypes:** atpSplitable; atpVariation **Tags:** atp.Splitkey=consistencyNeeds.shortName, consistencyNeeds.variationPoint.shortLabel vh.latestBindingTime=preCompileTime This Attribute is only used by the AUTOSAR Classic Platform.
port	PortPrototype	*	aggr	The `PortPrototype`s through which this `SwComponentType` can communicate. The aggregation of `PortPrototype` is subject to variability with the purpose to support the conditional existence of `PortPrototype`s. **Stereotypes:** atpSplitable; atpVariation **Tags:** atp.Splitkey=port.shortName, port.variationPoint.shortLabel vh.latestBindingTime=preCompileTime
portGroup	PortGroup	*	aggr	A port group being part of this component. **Stereotypes:** atpSplitable; atpVariation **Tags:** atp.Splitkey=portGroup.shortName, portGroup.variationPoint.shortLabel vh.latestBindingTime=preCompileTime
swcMappingConstraint	SwComponentMappingConstraints	*	ref	Reference to constraints that are valid for this SwComponentType. This Attribute is only used by the AUTOSAR Classic Platform.

swComponentDocumentation	SwComponentDocumentation	0..1	aggr	This adds a documentation to the `SwComponentType`.
				Stereotypes: atpSplitable; atpVariation
				Tags:
				atp.Splitkey=swComponentDocumentation, swComponentDocumentation.variationPoint.shortLabel
				vh.latestBindingTime=preCompileTime
				xml.sequenceOffset=-10

▽

△

Class	***SwComponentType*** (abstract)			
unitGroup	UnitGroup	*	ref	This allows for the specification of which `UnitGroup`s are relevant in the context of referencing `SwComponentType`. This Attribute is only used by the AUTOSAR Classic Platform.

Table B.15: SwComponentType

Class	**System**			
Note	The top level element of the System Description. The System description defines five major elements: Topology, Software, Communication, Mapping and Mapping Constraints. The System element directly aggregates the elements describing the Software, Mapping and Mapping Constraints; it contains a reference to an ASAM FIBEX description specifying Communication and Topology. **Tags:** atp.recommendedPackage=Systems			
Base	*ARElement*, *ARObject*, *AtpClassifier*, *AtpFeature*, *AtpStructureElement*, *CollectableElement*, *Identifiable*, *MultilanguageReferrable*, *PackageableElement*, *Referrable*, *UploadableDesignElement*, *UploadablePackageElement*			
Aggregated by	ARPackage.element, *AtpClassifier*.atpFeature			
Attribute	***Type***	***Mult.***	***Kind***	***Note***
clientId DefinitionSet	ClientIdDefinitionSet	*	ref	Set of Client Identifiers that are used for inter-ECU client-server communication in the System. This Attribute is only used by the AUTOSAR Classic Platform.
containerIPdu HeaderByte Order	ByteOrderEnum	0..1	attr	Defines the byteOrder of the header in ContainerIPdus. This Attribute is only used by the AUTOSAR Classic Platform.
ecuExtract Version	RevisionLabelString	0..1	attr	Version number of the Ecu Extract. This Attribute is only used by the AUTOSAR Classic Platform.
fibexElement	FibexElement	*	ref	Reference to ASAM FIBEX elements specifying Communication and Topology. All Fibex Elements used within a System Description shall be referenced from the System Element. atpVariation: In order to describe a product-line, all Fibex Elements can be optional. **Stereotypes:** atpSplitable; atpVariation **Tags:** atp.Splitkey=fibexElement.fibexElement, fibex Element.variationPoint.shortLabel vh.latestBindingTime=postBuild
interpolation Routine MappingSet	InterpolationRoutine MappingSet	*	ref	This reference identifies the InterpolationRoutineMapping Sets that are relevant in the context of the enclosing System. This Attribute is only used by the AUTOSAR Classic Platform.
j1939Shared AddressCluster	J1939SharedAddress Cluster	*	aggr	Collection of J1939Clusters that share a common address space for the routing of messages. **Stereotypes:** atpSplitable; atpVariation **Tags:** atp.Splitkey=j1939SharedAddressCluster.shortName, j1939SharedAddressCluster.variationPoint.shortLabel vh.latestBindingTime=postBuild This Attribute is only used by the AUTOSAR Classic Platform.

▽

△

Class	System			
mapping	SystemMapping	*	aggr	Aggregation of all mapping aspects (mapping of SW components to ECUs, mapping of data elements to signals, and mapping constraints). In order to support OEM / Tier 1 interaction and shared development for one common System this aggregation is atpSplitable and atpVariation. The content of System Mapping can be provided by several parties using different names for the SystemMapping. This element is not required when the System description is used for a network-only use-case. **Stereotypes:** atpSplitable; atpVariation **Tags:** atp.Splitkey=mapping.shortName, mapping.variation Point.shortLabel vh.latestBindingTime=postBuild
pncVector Length	PositiveInteger	0..1	attr	Length of the partial networking request release information vector (in bytes).
pncVectorOffset	PositiveInteger	0..1	attr	Absolute offset (with respect to the NM-PDU) of the partial networking request release information vector that is defined in bytes as an index starting with 0.
rootSoftware Composition	RootSwComposition Prototype	0..1	aggr	Aggregation of the root software composition, containing all software components in the System in a hierarchical structure. This element is not required when the System description is used for a network-only use-case. atpVariation: The RootSwCompositionPrototype can vary. **Stereotypes:** atpSplitable; atpVariation **Tags:** atp.Splitkey=rootSoftwareComposition.shortName, root SoftwareComposition.variationPoint.shortLabel vh.latestBindingTime=systemDesignTime This Attribute is only used by the AUTOSAR Classic Platform.
swCluster	CpSoftwareCluster	*	ref	CP Software Clusters of this System **Stereotypes:** atpSplitable; atpVariation **Tags:** atp.Splitkey=swCluster.cpSoftwareCluster, sw Cluster.variationPoint.shortLabel vh.latestBindingTime=systemDesignTime This Attribute is only used by the AUTOSAR Classic Platform.
systemCom SpecDefinition	SystemComSpec DefinitionSet	*	ref	Reference to the set of ComSpec definitions that are used for inter-ECU communication in the System.
system Documentation	Chapter	*	aggr	Possibility to provide additional documentation while defining the System. The System documentation can be composed of several chapters. **Stereotypes:** atpSplitable; atpVariation **Tags:** atp.Splitkey=systemDocumentation.shortName, system Documentation.variationPoint.shortLabel vh.latestBindingTime=systemDesignTime xml.sequenceOffset=-10 This Attribute is only used by the AUTOSAR Classic Platform.
systemVersion	RevisionLabelString	0..1	attr	Version number of the System Description.

Table B.16: System

Class	**VariableDataPrototype**			
Note	A `VariableDataPrototype` represents a formalized generic piece of information that is typically mutable by the application software layer. `VariableDataPrototype` is used in various contexts and the specific context gives the otherwise generic `VariableDataPrototype` a dedicated semantics.			
Base	*ARObject*, *AtpFeature*, *AtpPrototype*, *AutosarDataPrototype*, *DataPrototype*, *Identifiable*, *Multilanguage Referrable*, *Referrable*			
Aggregated by	ApplicationInterface.indication, *AtpClassifier*.atpFeature, BswInternalBehavior.arTypedPerInstance Memory, BswModuleDescription.providedData, BswModuleDescription.requiredData, BulkNvData Descriptor.bulkNvBlock, DiagnosticSovdAccessArgument.contentObject, *InternalBehavior*.staticMemory, NvBlockDescriptor.ramBlock, NvDataInterface.nvData, SenderReceiverInterface.dataElement, Service Interface.event, SwcInternalBehavior.arTypedPerInstanceMemory, SwcInternalBehavior.explicitInter RunnableVariable, SwcInternalBehavior.implicitInterRunnableVariable			
Attribute	***Type***	***Mult.***	***Kind***	***Note***
initValue	ValueSpecification	0..1	aggr	Specifies initial value(s) of the VariableDataPrototype

Table B.17: VariableDataPrototype