Document Title	System Template
Document Owner	AUTOSAR
Document Responsibility	AUTOSAR
Document Identification No	063
Document Classification	Standard

Document Version	4.2.0
Document Status	Final
Part of Release	4.0
Revision	3

Document Change History

Date	Version	Changed by	Description
08.11.2011	4.2.0	AUTOSAR Administration	• Added support for Partial Networking • Added support for Complex Device Drivers • Added support for new COM transfer properties • Added support for transmission mode switch via Com_SwitchIpduTxMode COM API • Added support for treating byte arrays with primitive type mapping • Added support for partial routing in signal gateways • Added support for FlexRay AUTOSAR TP • Added rules for creation of Pdu Triggerings and Pdu Ports • Explained the general approach of bit counting

— AUTOSAR CONFIDENTIAL —

22.10.2010	4.1.0	AUTOSAR Administration	• updated System class category names • Changed specification of PduLength parameter from bits to bytes • Made Flexray channel specific attributes optional • Clarified the usage of EcuPorts in System Extract/Ecu Extract • Allowed to define sending and receiving connections to EcuPorts for NmPdus, XcpPdus • Aligned FrTP model to AUTOSAR FrTp SWS • Replaced ComProcessingPeriod by three timebase parameters • Reworked E2E protection of selected I-PDUs • Corrected AssignFrameIdRange configuration in LIN model • Clarified the routing of ISignalGroups in the Signal Gateway
			• Extended the enumeration "TransferPropertyEnum" with the element "triggeredOnChange" • Added a subchapter to the appendix about special use cases that are supported by the System Template • Reworked SenderReceiverToSignalGroupMapping and ClientServerToSignalGroupMapping • Changed multiplicity between System and SystemMapping from 1 to 0..1.

04.12.2009	4.0.0	AUTOSAR Administration	• Implemented support for LIN 2.1 • Implemented support for Network Management (FlexRayNm, CanNm, LinNm, UdpNm) • adapted IPdu Multiplexer model to ASAM Fibex 3.1 • Reworked "ECU Extract" chapter • Introduced "System Extract" • Introduced EndToEndProtection for ISignalIPdus • Reworked "Transport Layer" chapter • Implemented Variant Handling concept • Implemented Documentation support concept • Implemented support for J1939 communication
			• Implemented support for TTCan • Implemented support for for TCP/IP and DoIP. • Introduced Pdu Counter and Pdu Replication • Implemented VMM/AMM concept • Introduced low-level routing of NPdu's • Implemented support for dynamic signals • Introduced PdurIPduGroups
26.01.2009	3.1.0	AUTOSAR Administration	• Clarified semantics of Data Mappings • Added inheritance from Identifiable to PduToFrameMapping • Added "FlexRayChannelName" attribute to FlexRayPhysicalChannel element.

03.06.2008	3.0.3	AUTOSAR Administration	• Added the boolean attribute "payloadPreambleIndicator" to the "FlexrayFrameTriggering". • Added extension that allows the assignment of IPduGroups to ECUs. • Added missing reference from "ClientServerComposite-TypeMapping" to "ArgumentPrototype" • Alignment with AUTOSAR IPduM SWS
15.02.2008	3.0.2	AUTOSAR Administration	• Legal disclaimer revised
31.01.2008	3.0.1	AUTOSAR Administration	• Moved "canAddressingMode" attribute from "CanCluster" to the "CanFrameTriggering" element • Clarified the descriptions of several elements and attributes.
04.12.2007	3.0.0	AUTOSAR Administration	• Communication part reworked from scratch • Alignment with ECU Configuration • Added support for Transport Protocols • Major changes in Topology chapter after harmonisation with Fibex (removed complex Topologies) • Document meta information extended • Small layout adaptations made

31.01.2007	2.0.0	AUTOSAR Administration	• Support for Signal Groups added. • Rework of the Topology Description • Introduction of PDUs. Description of the PDU Multiplexer, PDU Gateway. • FlexRay: multiple transmission of a frame within one communication cycle is supported now. • Removed the concept of Variant Descriptions (Properties) and CompToECUMappingConstraints relying on the property concept. • Split SwCompToEcuMapping in two classes in order to allow separation of SWC-to-ECU mapping and Implementation-to-SWC mapping. • Removed preliminary chapter on MOST as it is not part of the standard. • For all Instance References in the System Template added diagrams to the meta-model containing detailed representations of these references.
31.01.2007	2.0.0	AUTOSAR Administration	• Legal disclaimer revised • Release Notes added • "Advice for users" revised • "Revision Information" added
09.05.05	1.0.0	AUTOSAR Administration	Initial Release

Disclaimer

This specification 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 specification.

The material contained in this specification is protected by copyright and other types of Intellectual Property Rights. The commercial exploitation of the material contained in this specification requires a license to such Intellectual Property Rights.

This specification 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 specification may be utilized or reproduced, in any form or by any means, without permission in writing from the publisher.

The AUTOSAR specifications have been developed for automotive applications only. They have neither been developed, nor tested for non-automotive applications.

The word AUTOSAR and the AUTOSAR logo are registered trademarks.

Advice for users

AUTOSAR specifications may contain exemplary items (exemplary reference models, "use cases", and/or references to exemplary technical solutions, devices, processes or software).

Any such exemplary items are contained in the specifications for illustration purposes only, and they themselves are not part of the AUTOSAR Standard. Neither their presence in such specifications, nor any later documentation of AUTOSAR conformance of products actually implementing such exemplary items, imply that intellectual property rights covering such exemplary items are licensed under the same rules as applicable to the AUTOSAR Standard.

Table of Contents

— AUTOSAR CONFIDENTIAL —

— AUTOSAR CONFIDENTIAL —

References

[1] Generic Structure Template
AUTOSAR_TPS_GenericStructureTemplate.pdf

[2] Model Persistence Rules for XML
AUTOSAR_TR_XMLPersistenceRules.pdf

[3] Methodology
AUTOSAR_TR_Methodology.pdf

[4] Software Component Template
AUTOSAR_TPS_SoftwareComponentTemplate.pdf

[5] Specification of ECU Resource Template
AUTOSAR_TPS_ECUResourceTemplate.pdf

[6] Specification of Timing Extensions
AUTOSAR_TPS_TimingExtensions.pdf

[7] ASAM Fibex - Field Bus Exchange Format, Version 3.1
http://www.asam.net

[8] LIN Specification Package, Version 2.1
http://www.lin-subbus.org

[9] CAN specifications
http://www.can-cia.org

[10] MOST Specification, Version 2.5
http://www.mostnet.de

[11] FlexRay Protocol Specification
http://www.flexray.com

[12] Requirements on System Template
AUTOSAR_RS_SystemTemplate.pdf

[13] Layered Software Architecture
AUTOSAR_EXP_LayeredSoftwareArchitecture.pdf

[14] Specification of LIN Interface
AUTOSAR_SWS_LINInterface.pdf

[15] Specification of RTE Software
AUTOSAR_SWS_RTE.pdf

[16] Basic Software Module Description Template
AUTOSAR_TPS_BSWModuleDescriptionTemplate.pdf

[17] Specification of SW-C End-to-End Communication Protection Library
AUTOSAR_SWS_E2ELibrary.pdf

— AUTOSAR CONFIDENTIAL —

[18] Specification of I-PDU Multiplexer
AUTOSAR_SWS_IPDUMultiplexer.pdf

[19] Specification of Communication
AUTOSAR_SWS_COM.pdf

[20] Specification of ECU Configuration
AUTOSAR_TPS_ECUConfiguration.pdf

[21] ASAM MCD 2MC ASAP2 Interface Specification
http://www.asam.net
ASAP2-V1.51.pdf

[22] Software Process Engineering Meta-Model Specification
http://www.omg.org/spec/SPEM/2.0/

1 Introduction

1.1 Abbreviations

Abbreviation	*Meaning*
CAN	Controller Area Network
CAS	Collision Avoidance Symbol
CC	Communication Controller
DoIp	Diagnostics over IP
DTD	Document Type Definition
ECU	Electrical Control Unit
FIBEX	Field Bus Exchange Format
I^2C	Inter-Integrated Circuit
ID	Identifier
IPDU	Interaction Layer Protocol Data Unit
ISG	Inter-slot Gap
LIN	Local Interconnect Network
LPDU	Data Link Layer Protocol Data Unit
MOST	Media Oriented Systems Transport
NAD	Node Address for Diagnostic
NIT	Network Idle Time
NM	Network Management
NPDU	Network Layer Protocol Data Unit
OBD	Onboard Diagnostic
PDU	Protocol Data Unit
POC	Protocol Operation Control
RTE	Runtime Environment
SDU	Service Data Unit
SID	Service Identifier
SPI	Serial Peripheral Interface
SWC	Software Component
SWC-T	Software Component Template
SYS-T	System Template
TP	Transport Protocol
TTCAN	Time Triggered Controller Area Network
UML	Unified Modeling Language
VFB	Virtual Functional Bus
XML	Extensible Markup Language
XSD	XML Schema Definition

1.2 Methodology for Defining Formal Template

Figure 1.1 illustrates the overall methodology used to define formal templates. As is explained in the "Generic Structure Template" [1], it is important to separate a precise and concise model of the information that needs to be captured from the concrete XML-DTDs, XML-Schemas or other technology that is used to define the actual templates.

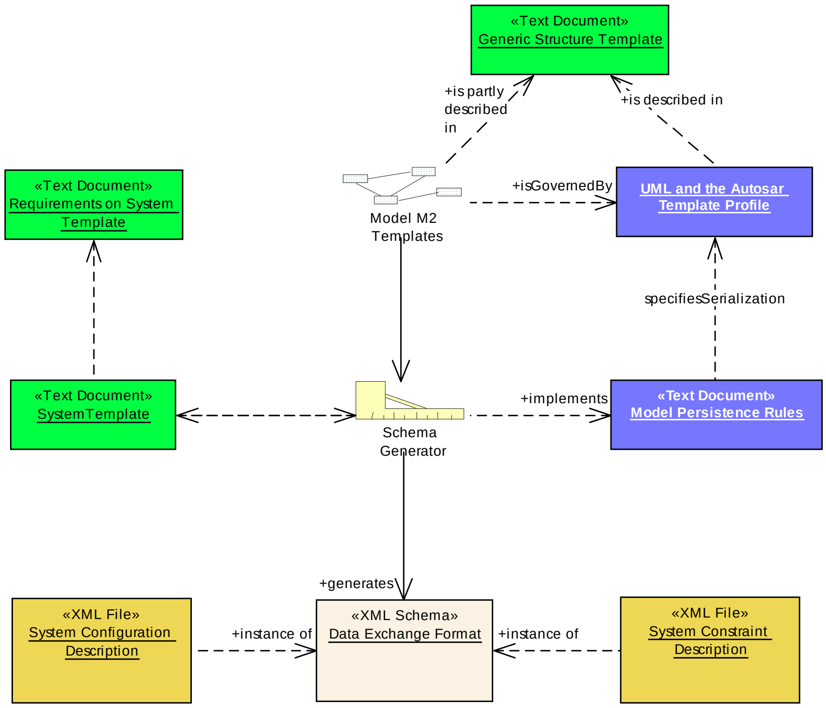

Figure 1.1: Methodology to define templates in AUTOSAR

The following documents describe the various aspects of the methodology:

1. The document called System Template (this document) describes the information that can be captured in the "system constraint" and "system configuration" description, independently from the mapping of this model on XML-technology. This document is based upon the AUTOSAR meta-model and contains an elaborate description of the semantics (the precise meaning) of all the information that can be captured within the relevant parts of this meta-model.

2. The UML and the AUTOSAR Template Profile [1] describes the basic concepts that should be used when creating content of the meta-model.

3. The document called "Model Persistence Rules for XML" [2] describes how XML is used and how the meta-model designed in the "System Template" should be translated by the "Schema Generator" (MMT) into XML-Schema (XSD) "Data Exchange Format". This "formalization strategy" is to be used for all data that is formally described in the meta-model. In particular this document is worth to read in order to understand the mapping of the meta-model and the XML based System template.

4. The "Generic Structure Template" [1] describes the top level structure which is common to all AUTOSAR templates and provides AUTOSAR standard mechanisms of modeling elements and patterns.
5. The concrete "Template", the "Data Exchange Format" is an XML schema which is generated out of the meta-model described in the "System Template" using the approach and the patterns defined in the "Model Persistence Rules for XML". This schema is typically used as input to tools. The M1-level system descriptions are XML files which can be validated against the schema. In that sense they are instances of the schema defining the XML representation of the template.

1.3 Scope

This document describes the system template and its use for the System Constraint Description and the System Configuration Description. In general a filled system template defines the relationship between the pure Software View on the System (represented by a top level SW Component Composition) and a Physical System Architecture with networked ECU instances. The system template is used in two stages of the "AUTOSAR Methodology" [3] (see Figure 1.2).

- As System Constraint Description it serves as input to the AUTOSAR system generator
- As System Configuration Description it defines the output of the AUTOSAR System Configuration Generator and serves as input to the AUTOSAR ECU Configuration Generator for the different ECUs defined in the description.
- As ECU Extract of the System Configuration Description it describes the ECU specific view on the System Description. It is individually generated for each of the System's ECU as the output of the AUTOSAR ECU Configuration Generator.

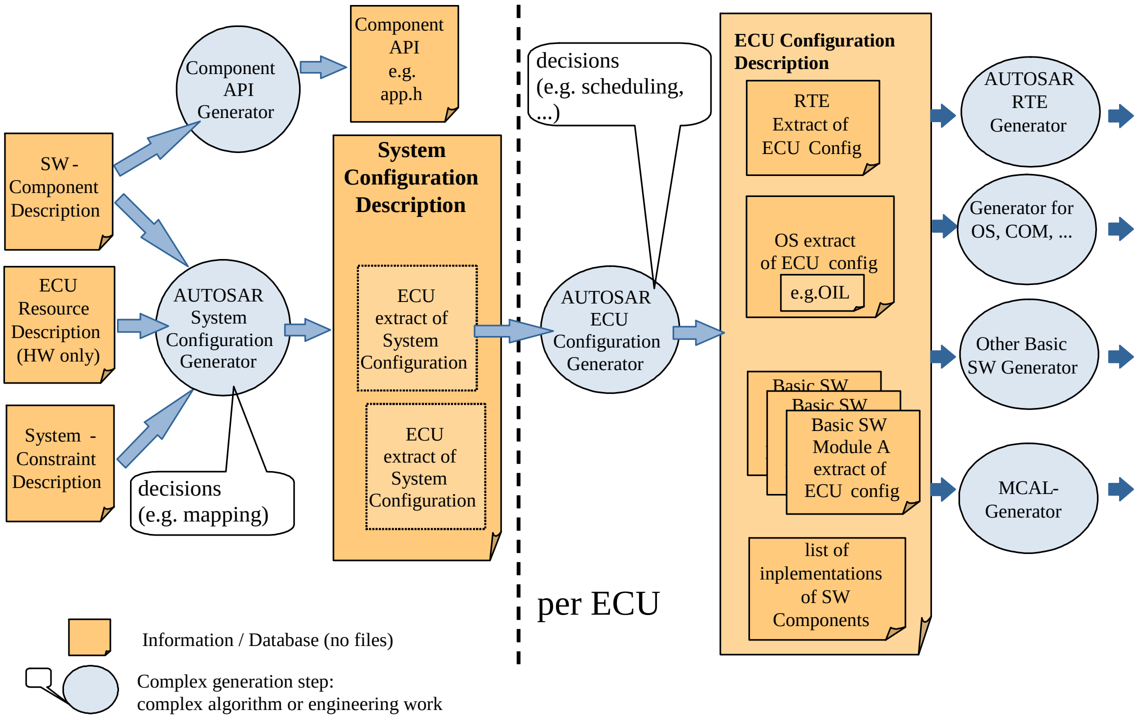

Figure 1.2: AUTOSAR Methodology

The System Template defines five major elements: Topology, Software, Communication, Mapping and Mapping Constraints, which will be defined in detail in the following chapters. Figure 1.3 gives an overview how these are used in the two different descriptions.

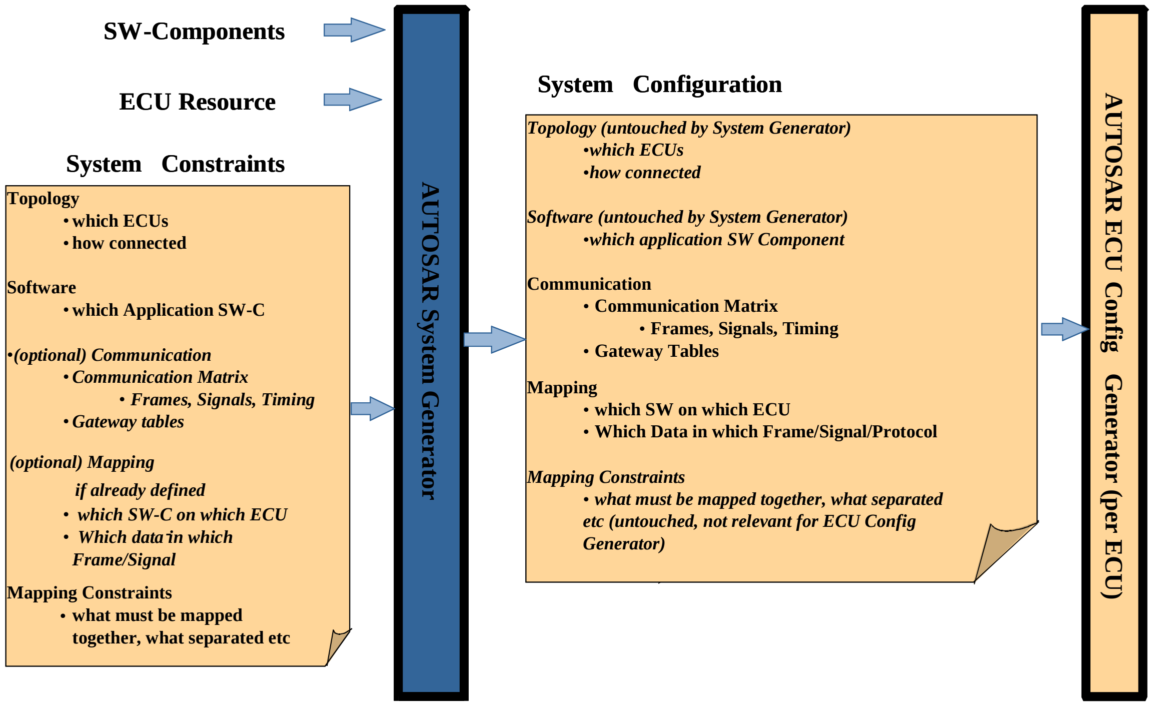

Figure 1.3: Scope of System Constraint Description and System Configuration Description

On Figure 1.3 some of the elements are marked `optional` for the System Constraint Description. If one starts with a new AUTOSAR project, these elements may not be present in the System Constraint Description. No (at least partial) functionality has been mapped yet, thus the communication matrix is not populated. But in most cases, many functional mappings are already predefined and contribute to the population of the communication matrix with their associated signals, thus being present in the System Constraint Description.

Reasons for such a predefinition are manifold. In some cases, hardware setup dictates where certain functionality resides, in some cases, a partial or complete communication matrix and/or completely configured ECUs (HW and SW) of another system (vehicle) has to be taken over. This approach is eased by the fact that System Configuration and System Constraint Description use the same format. That way it is possible to reuse parts of a System Configuration Description of the other system/vehicle in the actual System Constraint Description.

Furthermore, in the figure some of the elements are marked `untouched` for the System Configuration Description. This can have two reasons:

- The System Generator does not modify neither the Topology (networked ECUs) nor the Software, so these parts are just moved from System Constraint Description to System Configuration Description during the generation step.
- In a completed System Configuration Description, all SW components and all ECU-to-ECU communication have been mapped. Thus mapping constraints that limit the flexibility in the mapping phase of the system generator are obsolete

— AUTOSAR CONFIDENTIAL —

and will not be used in subsequent generator steps. They may however still be present for documentation and validation reasons.

Even if the communication matrix is determined as the result of the system configuration, the ECUs still have to be configured. This is done by the ECU configuration generator, which takes the System Configuration description as input and generates the ECU configuration description. The following guiding principles have been used to determine which information must be part of the System Configuration Description and which goes into the ECU Configuration Description:

- Information that is common for several ECUs and has to be agreed, must be part of the System Configuration Description and is thus covered by the System Template.
- Information, that only has ECU-local relevance is part of the ECU Configuration Description.

Thus the ECU Configuration Description will include the OS-schedule, the RTE-configuration and last but not least the configuration of the ECU basic software including the concrete communication drivers on that ECU.

1.4 UML Meta-Model

This chapter gives an overview of the AUTOSAR Unified Modeling Language (UML) meta-model. All AUTOSAR templates use a common meta-model. The templates describe software components, ECU resources, the Basic Software Modules, the ECU Configuration Parameters (ECU Configuration Description and ECU Configuration Parameter Definition) and the System.

The System Template defines all elements, their parameters and their relations, which are necessary for the System Constraint Description and the System Configuration Description.

— AUTOSAR CONFIDENTIAL —

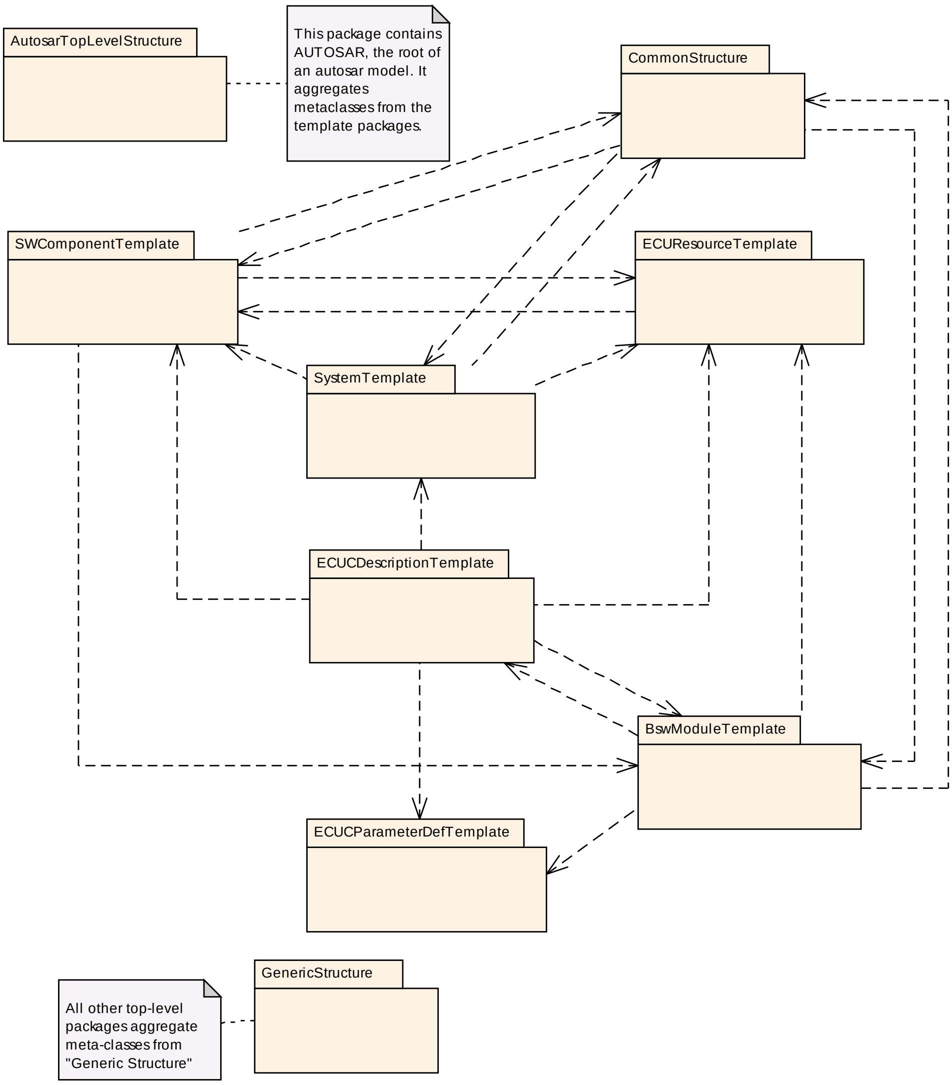

Figure 1.4: AUTOSAR Package Overview

Figure 1.4 shows the overall structure of the meta-model.

The dashed arrows in the diagram describe dependencies in terms of import-relationships between the packages within the meta-model. For example, the package `SystemTemplate` imports meta-classes defined in the packages `GenericStructure` [1], `SWComponentTemplate` [4] and `ECUResourceTemplate` [5].

For clarification, please note that the package `GenericStructure` contains some fundamental infrastructure meta-classes and common patterns that are described in [1]. As these are used by all other template specification the dependency associations are not depicted in the diagram for the sake of clarity.

Generic Structure provides details about

- Autosar Top level structure,
- Commonly used metaclasses and primitives

- Variant Handling
- Documentation

The ECU Resource Template deals with the description of the hardware resources of an ECU. The collection of all ECUs, which are integrated in the car, are described in the topology part of the System Configuration Description/System Constraint Description. Each of these ECUInstances uses the ECU Resource Template to describe the hardware resources. That's the reason, why the topology part has references to the ECU Resource Description.

The SW component description describes the SW components as well as their communication by data elements. The top-level software composition (`RootSwCompositionPrototype`) is part of the System Template (Software). This top-level software composition contains the functionality of the full system and describes the complete application software architecture of this system. The definition of the top level software composition uses the elements defined in the SW Component Template, like e.g. `SwComponentType`, `PortInterface`, `AssemblySwConnector` and `DelegationSwConnector`. That's why the System Description has references to the Software Component Description. The top level software composition is described in more detail in chapter 3.

Every template starts with an element `AUTOSAR`. While the models created in accordance to this guide are independent of the used formalization, it may still help the reader's understanding to note that `AUTOSAR` would also typically be the root element of a XML Schema generated from such a model. `AUTOSAR` can then contain one or more nested packages, simply allowing to further structure the contents of the M1 model[1].

[1] A model and its meta-model are said to be on different meta levels (also referred to as abstraction levels). In AUTOSAR a five layer meta-model hierarchy is used, consisting of the five meta levels M0, M1, M2, M3 and M4 where entities in M0 are expressed in terms of M1 entities, M1 is expressed in terms of M2 entities and so on. The AUTOSAR meta-model hierarchy is described in more detail in the Autosar Template Modeling Guide [1].

The top level element of the System Template is the class `System`, as shown in figure 1.5.

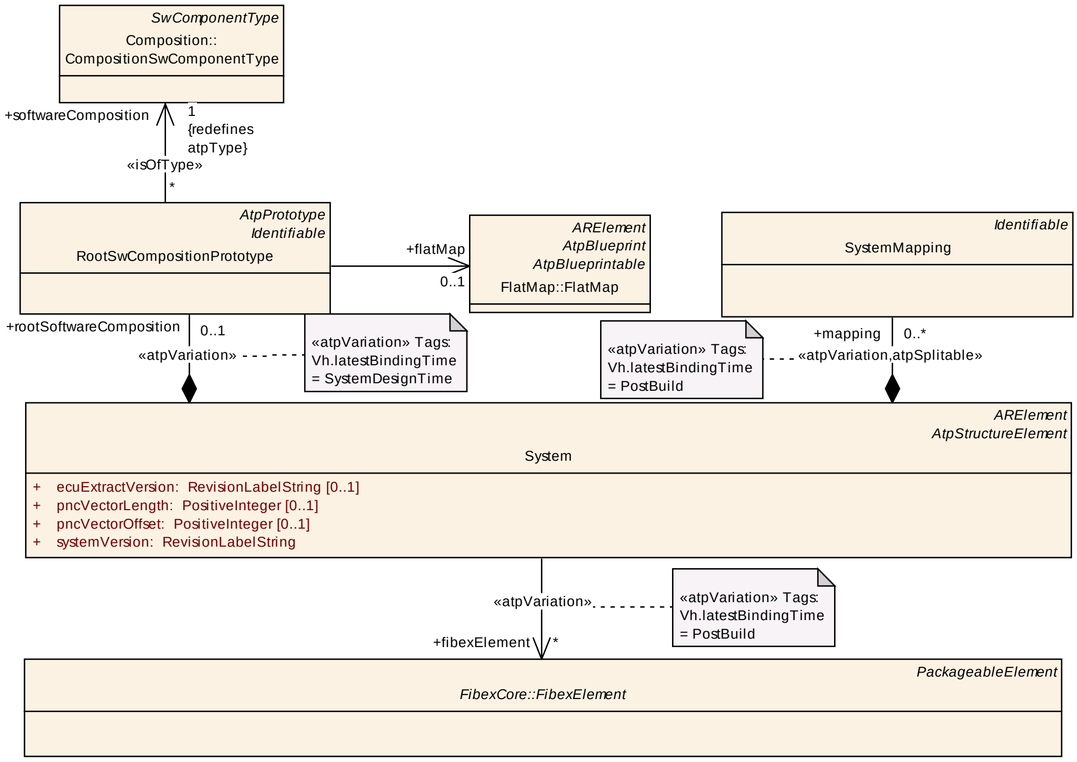

Figure 1.5: System Template Overview

`System` has relationships to all elements that define a system constraint description or system configuration description. It aggregates the `SystemMapping` and `Software Composition` elements. The `SystemMapping` area deals with mapping of software components to ECUs as well as with the mapping of data elements that are to be exchanged between software components onto signals and frames. The `RootSwCompositionPrototype` element contains a reference to the top level software composition.

The `System` class contains a reference to `FibexElements`. `FibexElements` can be defined in a stand alone and reusable way (hence they can simply be created in any package like ARElements), but on the other hand it shall be clear that a certain `FibexElement` actually belongs to a certain System Description. Thus, all `FibexElements` used within a System Description (i.e. contributing to the specification of the System communication and topology) shall be referenced from the `System` element. More details about the integration of FIBEX into the System Template will be given in chapter 1.5.

According to the different roles of the `System` class in the work products System Constraints, System Description or ECU Extract of System Description, the `category` attribute shall be used to clearly distinguish the intended usage in a specific work product.

Class	System			
Package	M2::AUTOSARTemplates::SystemTemplate			
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,Collectable Element,Identifiable,MultilanguageReferrable,PackageableElement,Referrable			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
ecuExtract Version	RevisionLabelSt ring	0..1	attr	Version number of the Ecu Extract.
fibexEleme nt	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 FibexElements can be optional. **Stereotypes:** atpVariation **Tags:** Vh.latestBindingTime=PostBuild
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 SystemMapping 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:** Vh.latestBindingTime=PostBuild atp.Splitkey=shortName, variationPoint.shortLabel
pncVector Length	PositiveInteger	0..1	attr	Length of the partial networking request release information vector.
pncVector Offset	PositiveInteger	0..1	attr	Absolute offset (with respect to the Frame) of the partial networking request release information vector.

Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
rootSoftwareComposition	RootSwCompositionPrototype	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:** atpVariation **Tags:** Vh.latestBindingTime=SystemDesignTime
systemDocumentation	Chapter	*	aggr	Possibility to provide additional documentation while defining the System. The System documentation can be composed of several chapters. **Stereotypes:** atpSplitable; atpVariation **Tags:** Vh.latestBindingTime=SystemDesignTime atp.Splitkey=shortName, VariationPoint.shortLabel xml.sequenceOffset=-10
systemVersion	RevisionLabelString	1	attr	Version number of the System Description.

Table 1.1: System

category	*Meaning*
SYSTEM_CONSTRAINTS	The `System` class is used to describe System Constraints. In this usage, it forms the core element of a System Constraints Description, serving as an input to the AUTOSAR System Generator.
SYSTEM_DESCRIPTION	The `System` class is used to describe the System Configuration of a complete AUTOSAR System. In this usage, it forms the core element of a System Description, the output of the AUTOSAR System Generator.
SYSTEM_EXTRACT	The `System` class is used to describe a subsystem specific view on the complete System Description. The System Extract is not fully decomposed and still contains compositions. The SYSTEM_EXTRACT is the basis for designing subsystems.
ECU_EXTRACT	The `System` class is used to describe the ECU specific view on the complete System Description. In this usage, it forms the core element of the ECU Extract, the output of the AUTOSAR ECU Configuration Extractor. The ECU Extract is fully decomposed and contains only atomic software components. The ECU Extract is the basis for setting up the ECU Configuration.

Table 1.2: System class categories

1.4.1 Meta-Model Tables

Beside the graphical visualization in UML diagrams, tables are used to specify the structure of the UML classes. In the following table one class is specified which holds an attribute and also a reference.

Class	**Class Name (Class names must be unique in the template model)**			
Package	Package that contains this class (Packages are a grouping mechanism for model elements)			
Note	class description			
Base	Name of the base class (When one class inherits from another, it is called a subclass and the class it inherits from is called a base class)			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
Attribute name	Integer	0..1	aggr	Attribute description
Role name	referenced class name	1..*	ref	Reference description

Table 1.3: Example of a class table

The headers in the table have the following meaning:

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

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

Note: The comment the modeler gave for the class. **Base Classes**: If applicable, the list of direct base classes.

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

Datatype: The datatype of an attribute of the class.

Multiplicity: The assigned multiplicity of the attribute, i.e. how many instances of the given datatype are associated with the attribute.

Kind: Specifies, whether the attributes is part of the class (`aggregation`) or just referenced by it (`reference`). Instance references are also indicated (`instanceRef`) in this field.

Note: The comment the modeler gave for the class attribute.

The stereotypes that can be used in the meta-model tables are described in the Generic Structure Template [1].

1.4.2 Detailed Representation of InstanceRef Associations

As a special type of association "instanceRef" refers to an exact instance of the referenced class, requiring additional information of the target and the context. This is explained in detail in the AUTOSAR Generic Structure Template [1]. Each "instanceRef" association can both be represented by the short form and by an detailed representation. For readability the diagrams in the main body of the specification use the short form. The detailed descriptions can be found in the Appendix C.

1.4.3 Variant Handling

The System Template supports the creation of Variants in many of its model elements. In the Metamodel all locations that may exhibit variability are marked with the stereotype `atpVariation`. This allows the definition of possible variation points. Tagged Values are used to specify additional informations.

There are four types of locations in the metamodel which may exhibit variability:

- Aggregations
- Associations
- Attribute Values
- Classes providing property sets

The reasons for the attachment of the stereotype `atpVariation` to certain model elements and the consequences for other model elements are explained in class tables in the following chapters. More details about the AUTOSAR Variant Handling Concept can be found in the AUTOSAR Generic Structure Template [1].

1.4.4 Timing Extensions

With AUTOSAR Release 4.0 a new set of concepts for the description and analysis of end-to-end timing constraints is introduced by the Specification of Timing Extensions. A subset of these extensions aims for the system level and can be used to enhance the descriptions that are already available in the System Template.

A dedicated description of the timing extensions that can be used at system level is given in chapter 3 (System timing) in the Specification of Timing Extensions [6].

1.4.5 Documentation Support

With AUTOSAR Release 4.0 the AUTOSAR XML schema provides support for integrated and well structured documentation. More details about the AUTOSAR Documentation Support concept can be found in the AUTOSAR Generic Structure Template [1]. An optional documentation block can be applied to any identifiable element. Furthermore, as shown in figure 1.6, the System Template provides the possibility of adding additional documentation to several non-identifiable elements. The documentation of a `System` is composed of several chapters.

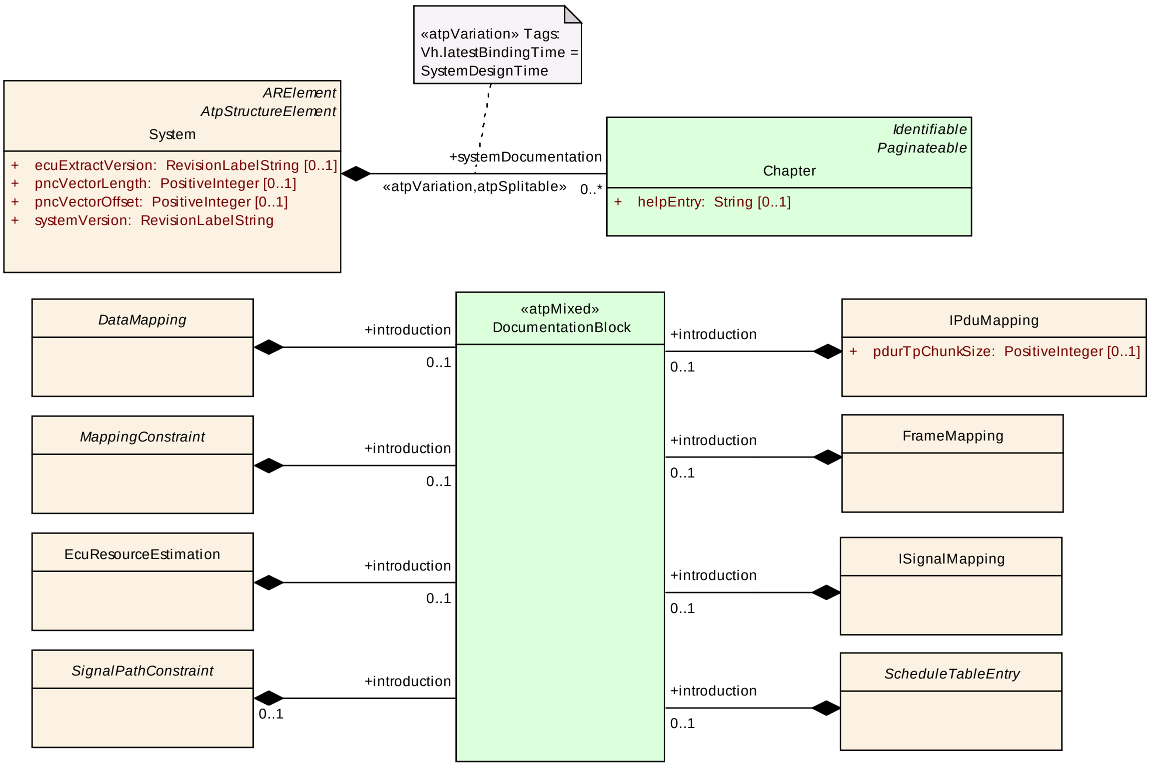

Figure 1.6: System Template Documentation Support

1.5 AUTOSAR System Template and ASAM FIBEX

FIBEX (Field Bus Exchange Format) [7] is an XML exchange format proposed for data exchange between tools that deal with bus communication Systems. The format supports the most common automotive data buses: LIN [8], CAN [9], MOST [10], FlexRay [11]. The covered areas of the exchange format are the functional network, system topology and the communication level. The functional network describes the software architecture of the system. In the system topology the logical layout of the system is described. This means it is documented which ECU is connected to which bus. The central purpose of a communication system is the exchange of frames with certain properties. The format is able to describe frames and their timing properties.

In future versions of the System Template a common subset between ASAM Fibex and Autosar will be harmonized. The current version of the System Template contains already the ASAM FIBEX description for communication and topology. Due to requirements of AUTOSAR some extensions were made to those descriptions. For instance the communication part is extended by a concept for PDUs (I-Pdus and N-Pdus). The harmonization between ASAM Fibex and AUTOSAR System Template is not finalized at this time.

In the UML Meta-Model the FIBEX contents are located in an own FIBEX UML Package. The top level `FibexElement` is referenced by the top level element `System` of the System Template. Similar to the usage of the `ARElement`, specializations of the

`FibexElement` represent elementary building blocks within the FIBEX package. Each of this elements will be described in more detail in the following chapters.

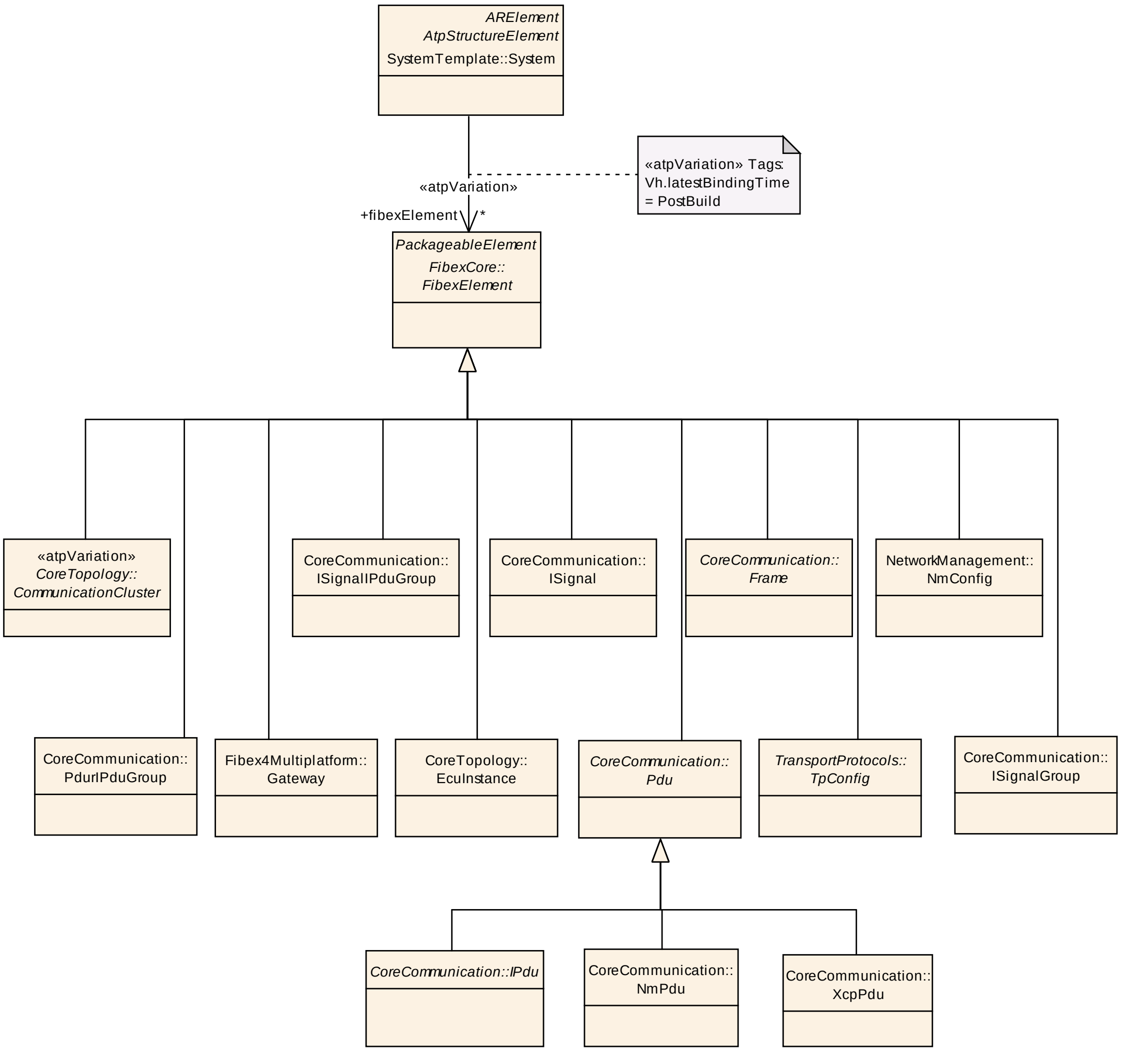

Figure 1.7: Fibex Elements

1.6 Document Conventions

Technical terms (Class Names) are typeset in monospaced font, e.g. `FrameTriggering`.

1.7 Requirements Tracing

The following table references the requirements specified in [12] and links to the fulfillment of these.

Requirement	Description	Satisfied by
[SYSCT0003]	No description	[TPS_SYST_1000]
[SYSCT0027]	No description	[TPS_SYST_1000]

1.8 Requirements not fulfilled by TPS requirements

This section contains a list of requirements that are not yet fulfilled by TPS requirements.

Requirement	Description	Satisfied by
SYSCT0001 Mixed Systems	System constraints, which arise through usage of mixed systems, must be treated by System Template	Definition of the communication matrix in the System Constraint Description can be made for any reason where it is necessary to restrict the system generator. One example is the usage of legacy ECUs in an AUTOSAR System. The frames that are transmitted or received by these legacy ECUs are constraints for the system generator because they cannot be changed, if the compatibility is supposed to be achieved without any changes at the legacy ECUs (chapter 7). In case that the System Description doesn't use a complete Software Component Description (VFB View) the swDataDefProps are used to configure the Data Semantics. (ISignal element in chapter 5.3).
SYSCT0002 Basic Software Resources and RTE Resources	The System Template has to cover resource requests of the basic SW and the RTE.	RTE and basic software resource estimations (chapter 4.3)
SYSCT0003 Iterative Development	During the development of an AUTOSAR system, solutions found in former steps of the system design process are themselves system constraints for the next system generation steps.	The system template is used in two stages of the AUTOSAR Methodology: System Constraint Description and System Configuration Description (chapter 1.3)

Requirement	Description	Satisfied by
SYSCT0006 Compatibility between the AUTOSAR Templates	The compatibility between the AUTOSAR Templates must be guaranteed. In this context, compatibility means that each AUTOSAR template can have references to elements of another AUTOSAR template.	Common UML Metamodel (chapter 1.4)
SYSCT0007 Mapping of Software Components to ECUs	The System Template has to describe the mapping of software components to ECUs. An optional mapping of software components to individual processing units residing in one ECU shall also be possible.	Software component Mapping (chapter 4.1)
SYSCT0008 SWC Clustering	The System Constraint Description has to cover the clustering of SW Components. SW Component Clustering means that two SW Components cannot be divided and must be mapped to the same ECU.	Software Component Mapping Constraints (chapter 4.1.3.1)
SYSCT0009 SWC Separation	The System Constraint Description has to cover the separation of SW Components. SW Component Separation means that two SW Components cannot be on the same ECU.	Software Component Mapping Constraints (chapter 4.1.3.2)
SYSCT0010 Exclusive Mapping of SW-C	The System Constraint Description has to cover the exclusion of SW-Cs from one or more ECUs. "Exclusion" means that the SW-C cannot be mapped to the ECUs it is excluded from. During the mapping process it can be useful to express that a specific SW-C cannot be mapped to one or more ECUs, based on ECU properties.	chapter 4.1.3.3 SwcToEcuMappingConstraint
SYSCT0011 Dedicated Mapping of SW-C	The System Constraint Description has to describe dedicated mapping of SW-Cs to one or more ECUs. "Dedicated mapping" means that the SW-C can only be mapped to the ECUs it is dedicated to. During the mapping process it can be useful to express that a specific SW-C can be only mapped to some ECUs, based on ECU properties.	chapter 4.1.3.3 SwcToEcuMappingConstraint
SYSCT0013 Topology	The System Template has to describe the topology of an EE System.	Topology (chapter 2)
SYSCT0014 Data Segmenting	The System Template must provide information, which can be used for the segmenting of (application) data to more than one frame.	In AUTOSAR, the Transport Layer has two main purposes: The segmentation and reassembly of messages that are too long to fit into one frame on the underlying communication cluster, and the re-use of fixed frame identifiers for different message content. (chapter 5.15 Transport Layer)
SYSCT0015 Bus bandwidth	The System Template shall support bandwidth calculation as a constraint for the definition of the Communication Matrix.	chapter Topology (2); Communication (chapter 5)
SYSCT0016 Dedicated physical connections	The System Constraint Description shall be able to describe that a signal has to be sent over a dedicated wire, which is only used by two SW-Components (sender and receiver).	Signal Path Constraint (chapter 4.2.2)

— AUTOSAR CONFIDENTIAL —

Requirement	Description	Satisfied by
SYSCT0017 Mapping of signals to the same physical line	The System Constraint Description shall be able to describe that a group of signals has to be sent via the same physical line.	common Signal Path (chapter 4.2.2.1)
SYSCT0018 Mapping of signals to different physical lines	The System Constraint Description shall be able to describe, if needed, that signals between ECUs are sent via different physical lines.	Separate Signal Path (chapter 4.2.2.4)
SYSCT0019 Mapping of signals to a specific physical line	The System Constraint Description shall be able to describe that signals have to be mapped to a specific physical line.	Permissible Signal Path (chapter 4.2.2.3)
SYSCT0020 Exclusion of signals from a specific physical line	The System Constraint Description shall be able to describe that signals have not to be mapped to a specific physical line.	Forbidden Signal Path (chapter 4.2.2.2)
SYSCT0021 ECU Communication via CAN	The System Template has to cover the system communication via CAN Bus.	CAN specific description (Topology and Communication)
SYSCT0022 ECU Communication via LIN	The System Template has to cover the system communication via LIN.	LIN specific description (Topology and Communication)
SYSCT0023 ECU Communication via MOST	The System Template has to cover the system communication via MOST.	not covered
SYSCT0024 ECU Communication via FlexRay	The System Template has to cover the system communication via FlexRay.	FlexRay specific description (Topology and Communication)
SYSCT0025 Derivation of ECU Configuration Parameters from the System Template	The System Template shall enable the configuration of the Com Stack of the ECU. It handles those parameters that are necessary to describe the inter-ECU communication. Configuration parameters local to an ECU are not in the scope of the System Template.	Harmonisation between Upstream Templates and ECU Configuration (chapter D)
SYSCT0026 Fibex compatibility	Whenever there is a considerable overlap between the System Template and the ASAM FIBEX Standard, the System Template shall adopt the structures of the ASAM FIBEX Standard.	AUTOSAR System Template and ASAM FIBEX (chapter 1.5)
SYSCT0027 ECU Extract	The ECU Extract is derived from a System Description. The specification for generating the ECU Extract shall be detailed enough to enable semantically unambiguous generation of this artifact.	ECU Extract (chapter 9)

Requirement	Description	Satisfied by
SYSCT0028 IPdu End-to-End Communication Protection support	The System Template shall enable to select E2E protection settings for IPdus.	"satisfied by Chapter 5.4.1"
SYSCT0029 Dynamic length signals	The System Template shall support a definition of dynamic length signals. A Signal shall have either a static length or its length should vary up to some statically defined maximum. Signals with a maximum length are called dynamic length signals.	(SystemSignal element in chapter 4.2)
SYSCT0030 Dynamic length IPdus	The System Template shall support a definition of IPdus that contain dynamic length signals.	It is possible to map dynamic length signals into an IPdu (Pdu element in chapter 5.4)
SYSCT0031 Distribution of Application and Vehicle Mode Requests	The System Template shall support the distribution of application and vehicle mode requests to all affected ECUs.	Mapping of ServiceProxySwComponentTypes to ECUs (chapter 4.1) and Data Mapping (chapter 4.2).
SYSCT0032 Topology Variants	The System Template shall provide the means to describe topology variants with optional/alternative ECUs and communication clusters.	chapter Variant Handling 1.4.3 and chapter Topology 2.
SYSCT0033 Software-to-ECU mapping variants	The System Template shall provide the means to describe alternative mappings of software components to ECUs.	chapter 1.4.3 Variant Handling and chapter 4.1 Software Component Mapping.
SYSCT0034 Timing variants	The System Template shall provide the means to describe alternative timing properties (e.g. trigger type, period, priority) and timing constraints (e.g. latency, age).	chapter 1.4.3 Variant Handling and chapter 5 Communication.
SYSCT0035 Data mapping variants	The System Template shall provide the means to describe data mapping Variants.	chapter 1.4.3 Variant Handling and chapter 4.2 Data Mapping.
SYSCT0036 Communication variants	The System Template shall provide the means to describe communication variants, such as alternative signal-to-PDU mappings, alternative communication paths, and alternative signal and PDU properties (e.g. data type, data length).	chapter 1.4.3 Variant Handling and chapter 5 Communication.
SYSCT0037 Timing Properties	The System Template shall provide the means to describe the timing properties of a system's dynamics, which are determined by the consumption of computation, communication, and other hardware resources.	chapter 1.4.4 Timing Extensions and chapter 5 Communication
SYSCT0038 Support of SAE J1939 Protocol Features	The System Template has to cover the system communication via SAE J1939.	SAE J1939 Protocol specific description (chapter 5.13)

Requirement	Description	Satisfied by
SYSCT0039 ECU Communication via Ethernet	The System Template has to cover the system communication via Ethernet.	Ethernet specific description (chapter 5.12)
SYSCT0040 Timing constraints	The System Template shall provide the means to describe the timing constraints of a system's dynamics, which are determined by the consumption of computation, communication, and other hardware resources.	Timing Extensions (chapter 1.4.4)
SYSCT0041 Variants in ECU Extract	The ECU Extract shall support variability of elements taken over or derived during the transformation from the System Description.	Variant Handling in ECU Extract (chapter 9.6)

2 Topology

This chapter explains how a vehicle's physical System Topology is being modeled in AUTOSAR (Example: Figure 2.1). A topology is formed by a number of ECUInstances that are interconnected to each other in order to form ensembles of ECUs and CommunicationClusters, which are further detailed by providing information on bus-specific properties.

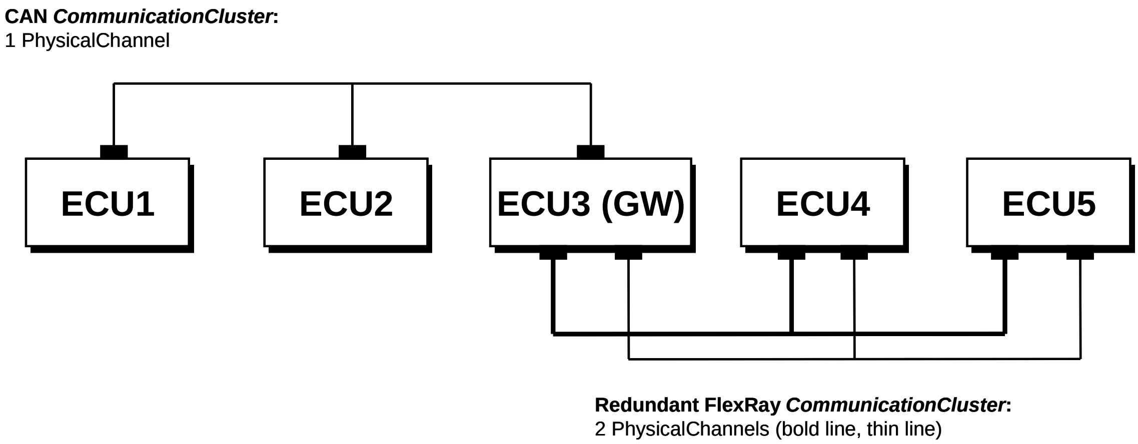

Figure 2.1: Example for a Communication Cluster within a physical network topology

In the AUTOSAR methodology [3] the topology description is one of the inputs for the System Generator. It serves as constraints for mapping the Software Components (see chapter 4.1) contained in the RootSwCompositionPrototype as well as for defining the System Communication matrix (see chapter 5). Gateways which allow the exchange of Signals between CommunicationClusters are covered in chapter 6.

2.1 ECUs and their communication capabilities

Within a System Topology, the ECUs actually being connected with each other are described in the form of ECUInstances. An ECUInstance needs to have one or more CommunicationController, the actual hardware device by means of which devices send and receive frames from the communication medium. Furthermore, the ECUInstance has one or more CommunicationConnectors which describe the bus interfaces of the ECUs and to specify the sending/receiving behavior.

In the AUTOSAR sense an ECU means a microcontroller plus peripherals and the according software/configuration. Therefore, each microcontroller requires its own ECU Configuration.

2.1.1 ECU Instance

`ECUInstance` describes the presence of a microcontroller in the vehicle. Within an `ECUInstance` class only those properties are described that are subject to system configuration; the actual description of the ECU hardware resources is done by the means of the ECU Resource Template [5]: It uses the `ECU` class and its aggregated hardware elements for defining a specific ECU type. The process of assigning an `ECU` type to `ECUInstance` is a mapping step (chapter 2.4.1) and performed latest in the System Generation step.

An `ECUInstance` can serve as a gateway if it is connected to two or more different clusters by two or more of its `CommunicationControllers`.

Class	**EcuInstance**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::FibexCore::CoreTopology			
Note	ECUInstances are used to define the ECUs used in the topology. The type of the ECU is defined by a reference to an ECU specified with the ECU resource description. **Tags:** atp.recommendedPackage=EcuInstances			
Base	ARObject,CollectableElement,FibexElement,Identifiable,Multilanguage Referrable,PackageableElement,Referrable			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
associatedComIPduGroup	ISignalIPduGroup	*	ref	With this reference it is possible to identify which ISignalIPduGroups are applicable for which CommunicationConnector/ ECU. Only top level ISignalIPduGroups shall be referenced by an EcuInstance. If an ISignalIPduGroup contains other ISignalIPduGroups than these contained ISignalIPduGroups shall not be referenced by the EcuInstance. Contained ISignalIPduGroups are associated to an EcuInstance via the top level ISignalIPduGroup.
associatedPdurIPduGroup	PdurIPduGroup	*	ref	With this reference it is possible to identify which PduR IPdu Groups are applicable for which CommunicationConnector/ ECU.
canTpAddress	CanTpAddress	*	ref	A Tp Address can be assigned to an ECU without an existing TP Configuration. If TpNodes are described this reference shall not be used.
comConfigurationGwTimeBase	TimeValue	0..1	attr	The period between successive calls to Com_MainFunctionRouteSignals of the AUTOSAR COM module in seconds.
comConfigurationRxTimeBase	TimeValue	0..1	attr	The period between successive calls to Com_MainFunctionRx of the AUTOSAR COM module in seconds.
comConfigurationTxTimeBase	TimeValue	0..1	attr	The period between successive calls to Com_MainFunctionTx of the AUTOSAR COM module in seconds.

Attribute	**Datatype**	**Mul.**	**Kind**	**Note**
comEnableMDTForCyclicTransmission	Boolean	0..1	attr	Enables for the Com module of this EcuInstance the minimum delay time monitoring for cyclic and repeated transmissions (TransmissionModeTiming has cyclicTiming assigned or eventControlledTiming with numberOfRepetitions > 0).
commController	CommunicationController	1..*	aggr	CommunicationControllers of the ECU.
connector	CommunicationConnector	*	aggr	All channels controlled by a single controller.
diagnosticAddress	Integer	0..1	attr	An ECU specific ID for responses of diagnostic routines.
partition	EcuPartition	*	aggr	Optional definition of Partitions within an Ecu.
sleepModeSupported	Boolean	1	attr	Specifies whether the ECU instance may be put to a "low power mode" TRUE: sleep mode is supported FALSE: sleep mode is not supported Note: This flag may only be set to TRUE if the feature is supported by both hardware and basic software.
tpAddress	TpAddress	*	ref	A Tp Address can be assigned to an ECU without an existing TP Configuration. If TpNodes are described this reference shall not be used.
wakeUpOverBusSupported	Boolean	1	attr	Driver support for wakeup over Bus.

Table 2.1: EcuInstance

Class	**EcuPartition**			
Package	M2::AUTOSARTemplates::SystemTemplate::SWmapping			
Note	Partitions are used as error containment regions. They permit the grouping of SWCs and resources and allow to describe recovery policies individually for each partition. Partitions can be terminated or restarted during run-time as a result of a detected error.			
Base	ARObject,Identifiable,MultilanguageReferrable,Referrable			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**
execInUserMode	Boolean	1	attr	A partition can execute either in CPU user mode (execInUserMode = TRUE) or supervisor mode (execInUserMode = FALSE). In user mode, the partition has a limited access to memory, to memory mapped hardware and to CPU. In user mode, the partition is mapped to a non-trusted OS-Application.

Table 2.2: EcuPartition

— AUTOSAR CONFIDENTIAL —

2.1.2 Communication Controller

A `CommunicationController` is a dedicated hardware device by means of which hosts are sending frames to and receiving frames from the communication medium.

In order to illustrate the relationship of an CommunicationController to the AUTOSAR `CommunicationPeripheral` defined in the ECU Resource Description, a mapping between these two classes may be specified using the `CommunicationControllerMapping` (see chapter 2.4.2).

Class	≪**atpVariation**≫ **CommunicationController (abstract)**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::FibexCore::CoreTopology			
Note	The communication controller is a dedicated hardware device by means of which hosts are sending frames to and receiving frames from the communication medium. **Tags:** Vh.latestBindingTime=PostBuild			
Base	ARObject,Identifiable,MultilanguageReferrable,Referrable			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
–	–	–	–	–

Table 2.3: CommunicationController

2.1.3 Communication Connector

An ECUInstance uses CommunicationConnector elements in order to describe its bus interfaces and to specify the sending/receiving behavior.

CommunicationConnector connects the ECUInstance it is associated with to the PhysicalChannel (see chapter 2.2.2), using the CommunicationController it references, realizing it. The reference towards CommunicationController is optional, so ECUInstances can be assigned to channels even before the controller is defined.

Class	**CommunicationConnector (abstract)**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::FibexCore::CoreTopology			
Note	The connection between the referencing ECU and the referenced channel via the referenced controller. Connectors are used to describe the bus interfaces of the ECUs and to specify the sending/receiving behavior. Each CommunicationConnector has a reference to exactly one communicationController. The communicationController can be referenced by several CommunicationConnector elements. This is important for the FlexRay Bus. FlexRay communicates via two physical channels. But only one controller in an ECU is responsible for both channels. Thus, two connectors (for channel A and for channel B) must reference to the same controller. Note: Several CommunicationConnectors can be assigned to one PhysicalChannel in the scope of one ECU Instance.			
Base	ARObject,Identifiable,MultilanguageReferrable,Referrable			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**
commController	CommunicationController	1	ref	Reference to the communication controller. The CommunicationConnector and referenced CommunicationController must be aggregated by the same ECUInstance.
ecuCommPortInstance	CommConnectorPort	*	aggr	An ECUs reception or send ports. atpVariation: If signals/PDUs/frames are variable, the corresponding ports must be variable, too. **Stereotypes:** atpVariation **Tags:** Vh.latestBindingTime=PostBuild
pncGatewayType	PncGatewayTypeEnum	0..1	attr	Defines if this EcuInstance shall implement the PncGateway functionality on this CommunicationConnector and its respective PhysicalChannel. Several EcuInstances on the same PhysicalChannel can have the PncGateway functionality enabled, but only one of them shall have the pncGatewayType "active".

Table 2.4: CommunicationConnector

Enumeration	**PncGatewayTypeEnum**
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::FibexCore::CoreTopology
Note	Defines the PncGateway roles.
Literal	***Description***
active	The active PncGateway functionality shall be performed
none	No PncGateway functionality shall be performed
passive	The passive PncGateway functionality shall be performed

Table 2.5: PncGatewayTypeEnum

Note: Use-case for the relation of several `CommunicationConnectors` assigned to one `PhysicalChannel` in the scope of one `ECUInstance`: One safety measure for a safety relevant ECU can be to have two transceivers (and two controllers) connected to the same network (Bus). In case a safety violation is detected one transceiver can be disabled and the respective Frames are blocked. The other transceiver stays active and keeps the ECU alive for diagnostics.

2.2 Communication Clusters

`ECUInstances` are linked together by a communication medium of arbitrary topology (bus, star, ring, tree) in order to form a `CommunicationCluster`. It aggregates one or more `PhysicalChannels`, representing the communication medium. Depending on the communication standard, a `CommunicationCluster` may have exactly one or more (redundant) `PhysicalChannels`.

An `ECUInstance` is included into the communication cluster by having the `ECUInstance's CommunicationConnector` reference to the `PhysicalChannel` it is connected to.

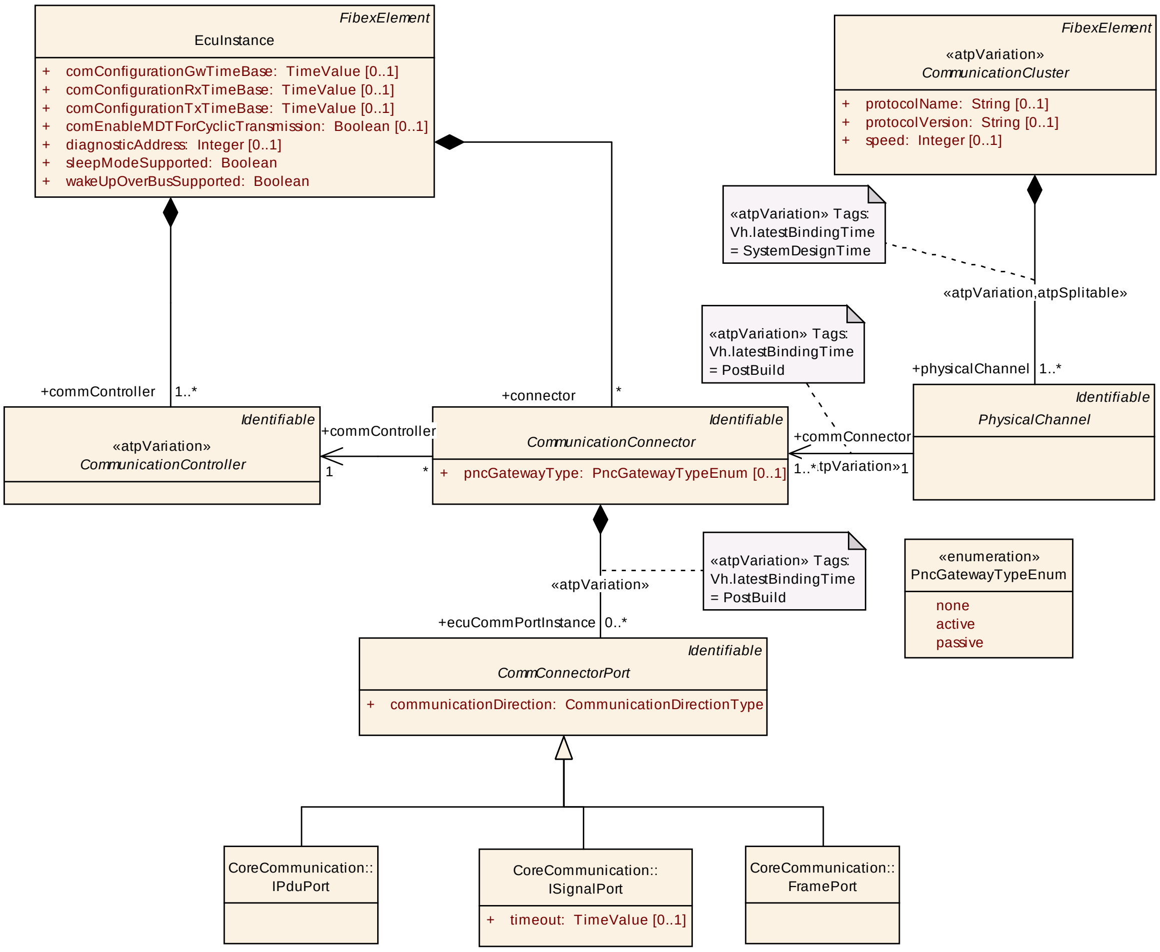

Figure 2.2: Topology elements (Topology)

[constr_3008] **`EcuInstance` subelements** ⌈The `CommunicationConnector` and the `CommunicationController` that is referenced by the `CommunicationConnector` must be owned by the same `ECUInstance`. ⌋

2.2.1 Communication Cluster

The `CommunicationCluster` is the main element to describe the topological connection of communicating ECUs. These are linked into an ensemble by a communication medium of arbitrary topology (bus, star, ring, tree). A `CommunicationCluster` aggregates one or more `PhysicalChannels` representing the communication medium. All ECUs within a CommunicationCluster communicate within the same address range. Note that the same ECU can participate in more than one `CommunicationCluster` if it has more than one `CommunicationConnector` being connected to different clusters' `PhysicalChannels`.

Class	≪atpVariation≫ **CommunicationCluster (abstract)**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::FibexCore::CoreTopology			
Note	The CommunicationCluster is the main element to describe the topological connection of communicating ECUs. A cluster describes the ensemble of ECUs, which are linked by a communication medium of arbitrary topology (bus, star, ring, ...). The nodes within the cluster share the same communication protocol, which may be event-triggered, time-triggered or a combination of both. A CommunicationCluster aggregates one or more physical channels. All physical channels that are aggregated by a communication cluster are synchronized with each other. **Tags:** Vh.latestBindingTime=PostBuild			
Base	ARObject,CollectableElement,FibexElement,Identifiable,Multilanguage Referrable,PackageableElement,Referrable			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**
physicalChannel	PhysicalChannel	1..*	aggr	This relationship defines which channel element belongs to which cluster. A channel must be assigned to exactly one cluster, whereas a cluster may have one or more channels. **Stereotypes:** atpSplitable; atpVariation **Tags:** Vh.latestBindingTime=SystemDesignTime atp.Splitkey=shortName, variationPoint.shortLabel
protocolName	String	0..1	attr	The name of the protocol used.
protocolVersion	String	0..1	attr	The version of the protocol used.
speed	Integer	0..1	attr	Channels speed in kbps.

Table 2.6: CommunicationCluster

Some communication clusters need, additional to the general attributes which are valid for all communication clusters, specialized attributes to describe the individual communication cluster properties. The bustype-specific specializations of `CommunicationCluster` (Figure 2.3) are further detailed in chapter 2.3.

2.2.2 Physical Channel

`PhysicalChannel` represents the communication medium that is used to send and receive information between two communicating ECUs. Each `CommunicationCluster` has at least one `PhysicalChannel`. Bus systems like CAN and LIN have exactly one `PhysicalChannel`. A FlexRay cluster may have more than one PhysicalChannel that can be used in parallel for redundant communication.

Class	PhysicalChannel (abstract)			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::FibexCore::CoreTopology			
Note	A physical channel is the transmission medium that is used to send and receive information between two communicating ECUs. Each CommunicationCluster has at least one physical channel. Bus systems like CAN and LIN only have exactly one PhysicalChannel. A FlexRay cluster may have more than one PhysicalChannels that may be used in parallel for redundant communication. An ECU is part of a cluster if it contains at least one controller that is connected to at least one channel of the cluster.			
Base	ARObject,Identifiable,MultilanguageReferrable,Referrable			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**
commConnector	CommunicationConnector	1..*	ref	Reference to the ECUInstance to which the channel is connected. atpVariation: Variable assignment of Physical Channels to different CommunicationConnectors is expressed with this variation. **Stereotypes:** atpVariation **Tags:** Vh.latestBindingTime=PostBuild
frameTriggering	FrameTriggering	*	aggr	One frame triggering is defined for exactly one channel. Channels may have assigned an arbitrary number of frame triggerings. atpVariation: If signals/PDUs/frames are variable, the corresponding triggerings must be variable, too. **Stereotypes:** atpSplitable; atpVariation **Tags:** Vh.latestBindingTime=PostBuild atp.Splitkey=shortName, variationPoint.shortLabel
iSignalTriggering	ISignalTriggering	*	aggr	One ISignalTriggering is defined for exactly one channel. Channels may have assigned an arbitrary number of ISignaltriggerings. atpVariation: If signals/PDUs/frames are variable, the corresponding triggerings must be variable, too. **Stereotypes:** atpSplitable; atpVariation **Tags:** Vh.latestBindingTime=PostBuild atp.Splitkey=shortName, variationPoint.shortLabel
pduTriggering	PduTriggering	*	aggr	One PduTriggering is defined for exactly one channel. Channels may have assigned an arbitrary number of I-Pdu triggerings. atpVariation: If signals/PDUs/frames are variable, the corresponding triggerings must be variable, too. **Stereotypes:** atpSplitable; atpVariation **Tags:** Vh.latestBindingTime=PostBuild atp.Splitkey=shortName, variationPoint.shortLabel

— AUTOSAR CONFIDENTIAL —

Attribute	Datatype	Mul.	Kind	Note

Table 2.7: PhysicalChannel

2.3 Specialized Attributes of the Topology Entities

According to their characteristic features, different communication standards like FlexRay, CAN, TTCAN, LIN and Ethernet have individual attributes that need to be described additionally to the common topology classes. Figure 2.3 shows the specialization of the `CommunicationCluster` into the more specific `FlexrayCluster`, `CanCluster`, `TtcanCluster`, `LinCluster` and `EthernetCluster`.

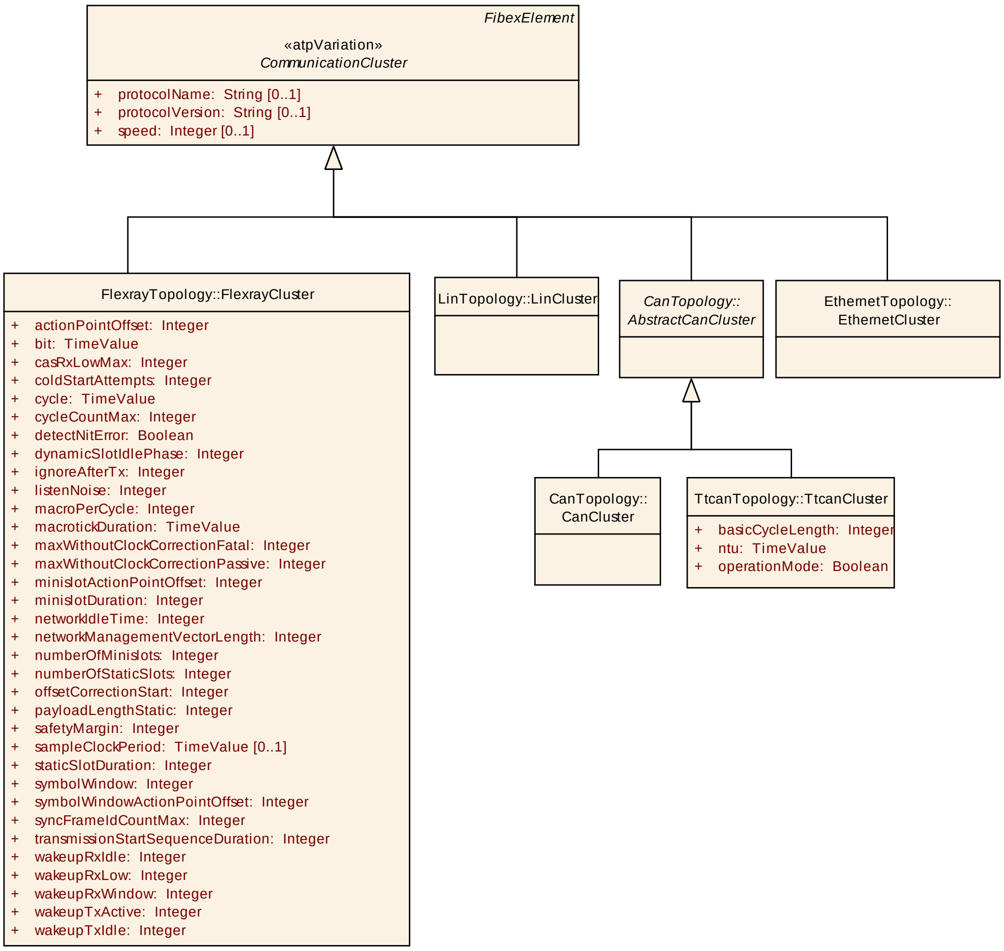

Figure 2.3: Specialized `CommunicationCluster` attributes (TopologyAttributeRefinement)

2.3.1 CAN

Modeling of the CAN bus is supported in the System Template by the means of four specialized meta-model classes: `CanCluster`, `CanCommunicationController`, `CanPhysicalChannel`, `CanCommunicationConnector` (Figure 2.4).

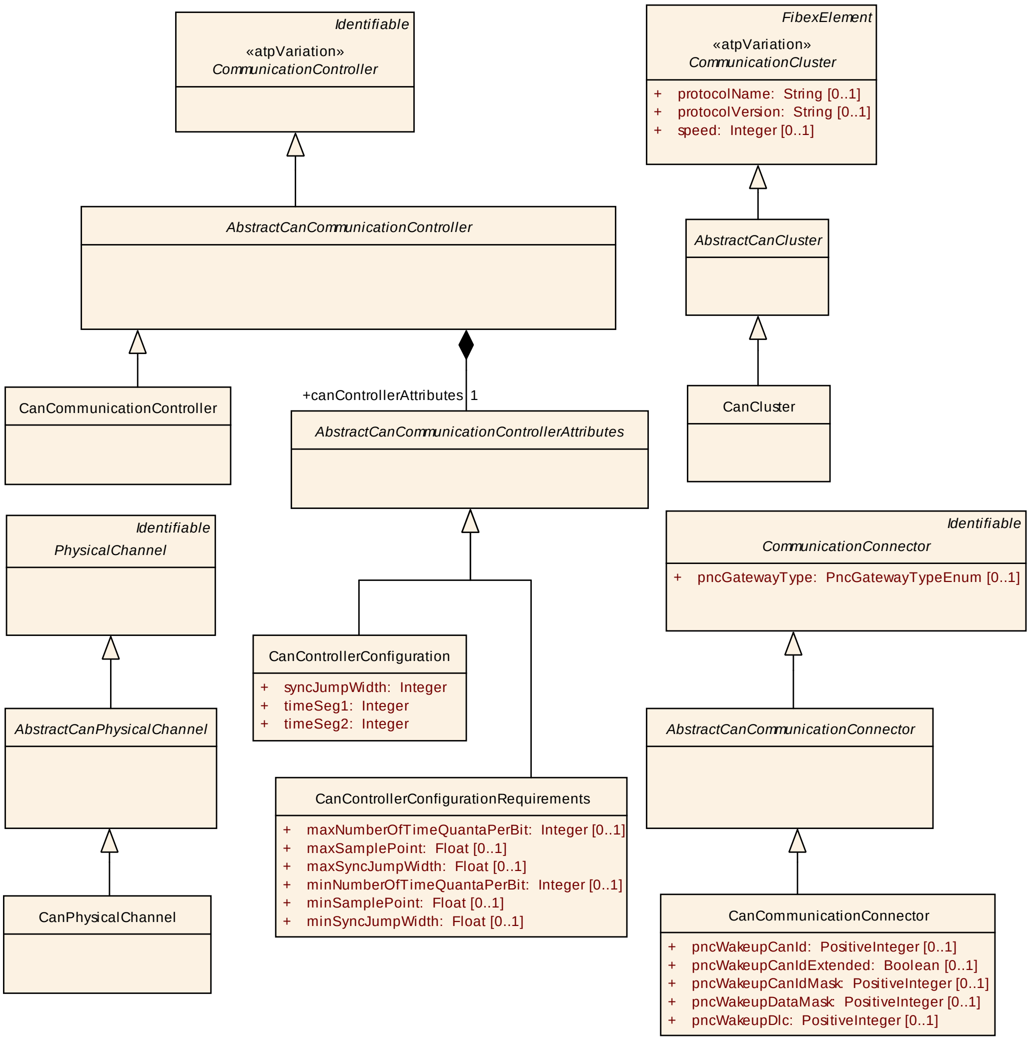

Figure 2.4: CAN bus elements (Fibex4Can_Topology)

2.3.1.1 CAN Cluster

`CanCluster` specifies the existence of a CAN cluster in the system's physical topology. It contains additional CAN-specific cluster-wide attributes. The common CAN and TTCAN attributes are collected in the `AbstractCanCluster` class.

Class	≪atpVariation≫ **AbstractCanCluster (abstract)**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Can::CanTopology			
Note	Abstract class that is used to collect the common TtCAN and CAN Cluster attributes.			
Base	ARObject,CollectableElement,CommunicationCluster,Fibex Element,Identifiable,MultilanguageReferrable,PackageableElement,Referrable			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**
–	–	–	–	–

Table 2.8: AbstractCanCluster

Class	≪atpVariation≫ **CanCluster**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Can::CanTopology			
Note	CAN bus specific cluster attributes. **Tags:** atp.recommendedPackage=CommunicationClusters			
Base	ARObject,AbstractCanCluster,CollectableElement,CommunicationCluster,Fibex Element,Identifiable,MultilanguageReferrable,PackageableElement,Referrable			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**
–	–	–	–	–

Table 2.9: CanCluster

2.3.1.2 CAN Communication Controller

`CanCommunicationController` is a specialization of the abstract `CommunicationController` class. It contains the specific CAN controller attributes needed for configuring the CAN stack in an ECU connected to a certain CAN cluster. The common CAN and TTCAN attributes are collected in the `AbstractCanCommunicationController` class. It is possible to specify the CAN Controller configuration parameters as exact values or as requirements that have to be respected by the ECU developer. Therefore the two elements `CanControllerConfiguration` and `CanControllerConfigurationRequirements` were created.

Class	≪atpVariation≫ **CanCommunicationController**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Can::CanTopology			
Note	CAN bus specific communication port attributes.			
Base	ARObject,AbstractCanCommunicationController,Communication Controller,Identifiable,MultilanguageReferrable,Referrable			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**
–	–	–	–	–

Table 2.10: CanCommunicationController

— AUTOSAR CONFIDENTIAL —

Class	≪atpVariation≫ **AbstractCanCommunicationController (abstract)**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Can::CanTopology			
Note	Abstract class that is used to collect the common TtCAN and CAN Controller attributes.			
Base	ARObject,CommunicationController,Identifiable,MultilanguageReferrable,Referrable			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**
canControllerAttributes	AbstractCanCommunicationControllerAttributes	1	aggr	CAN Bit Timing configuration

Table 2.11: AbstractCanCommunicationController

Class	**AbstractCanCommunicationControllerAttributes (abstract)**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Can::CanTopology			
Note	For the configuration of the CanController parameters two different approaches can be used: 1. Providing exact values which are taken by the ECU developer (CanControllerConfiguration). 2. Providing ranges of values which are taken as requirements and have to be respected by the ECU developer (CanControllerConfigurationRequirements).			
Base	ARObject			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**
–	–	–	–	–

Table 2.12: AbstractCanCommunicationControllerAttributes

Class	**CanControllerConfiguration**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Can::CanTopology			
Note	This element is used for the specification of the exact CAN Bit Timing configuration parameter values.			
Base	ARObject,AbstractCanCommunicationControllerAttributes			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**
syncJumpWidth	Integer	1	attr	The number of quanta in the Synchronization Jump Width, SJW. The (Re-)Synchronization Jump Width (SJW) defines how far a resynchronization may move the Sample Point inside the limits defined by the Phase Buffer Segments to compensate for edge phase errors.
timeSeg1	Integer	1	attr	The number of quanta before the sampling point. The propagation time segment is factored into the timeSeg1 configuration parameter: timeSeg1 = tPROP_SEG + tPHASE_SEG1
timeSeg2	Integer	1	attr	The number of quanta after the sampling point: timeSeg2 = Phase_Seg2

Table 2.13: CanControllerConfiguration

Class	**CanControllerConfigurationRequirements**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Can::CanTopology			
Note	This element allows the specification of ranges for the CAN Bit Timing configuration parameters. These ranges are taken as requirements and have to be respected by the ECU developer.			
Base	ARObject,AbstractCanCommunicationControllerAttributes			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**
maxNumberOfTimeQuantaPerBit	Integer	0..1	attr	Maximum number of time quanta in the bit time.
maxSamplePoint	Float	0..1	attr	The max. value of the sample point as a percentage of the total bit time.
maxSyncJumpWidth	Float	0..1	attr	The max. Synchronization Jump Width value as a percentage of the total bit time. The (Re-)Synchronization Jump Width (SJW) defines how far a resynchronization may move the Sample Point inside the limits defined by the Phase Buffer Segments to compensate for edge phase errors.
minNumberOfTimeQuantaPerBit	Integer	0..1	attr	Minimum number of time quanta in the bit time.
minSamplePoint	Float	0..1	attr	The min. value of the sample point as a percentage of the total bit time.
minSyncJumpWidth	Float	0..1	attr	The min. Synchronization Jump Width value as a percentage of the total bit time. The (Re-)Synchronization Jump Width (SJW) defines how far a resynchronization may move the Sample Point inside the limits defined by the Phase Buffer Segments to compensate for edge phase errors.

Table 2.14: CanControllerConfigurationRequirements

2.3.1.3 CAN Physical Channel

CanPhysicalChannel is a specialization of the abstract PhysicalChannel class. It contains the specific CAN PhysicalChannel attributes. The common CAN and TTCAN attributes are collected in the AbstractCanPhysicalChannel class.

Class	**AbstractCanPhysicalChannel (abstract)**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Can::CanTopology			
Note	Abstract class that is used to collect the common TtCAN and CAN PhysicalChannel attributes.			
Base	ARObject,Identifiable,MultilanguageReferrable,PhysicalChannel,Referrable			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**
–	–	–	–	–

Table 2.15: AbstractCanPhysicalChannel

Class	**CanPhysicalChannel**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Can::CanTopology			
Note	CAN bus specific physical channel attributes.			
Base	ARObject,AbstractCanPhysicalChannel,Identifiable,MultilanguageReferrable,Physical Channel,Referrable			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**
–	–	–	–	–

Table 2.16: CanPhysicalChannel

[constr_3003] Number of CAN channels ⌈CAN clusters shall aggregate exactly one PhysicalChannel.⌋

2.3.1.4 CAN Communication Connector

CanCommunicationConnector is a specialization of the abstract Communication-Connector class. It contains the specific CAN CommunicationConnector attributes. The common CAN and TTCAN attributes are collected in the Abstract-CanCommunicationConnector class.

Class	**AbstractCanCommunicationConnector (abstract)**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Can::CanTopology			
Note	Abstract class that is used to collect the common TtCAN and CAN CommunicationConnector attributes.			
Base	ARObject,CommunicationConnector,Identifiable,MultilanguageReferrable,Referrable			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**
–	–	–	–	–

Table 2.17: AbstractCanCommunicationConnector

— AUTOSAR CONFIDENTIAL —

Class	**CanCommunicationConnector**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Can::CanTopology			
Note	CAN bus specific communication connector attributes.			
Base	ARObject,AbstractCanCommunicationConnector,Communication Connector,Identifiable,MultilanguageReferrable,Referrable			
Attribute	*Datatype*	*Mul.*	*Kind*	*Note*
pncWakeupCanId	PositiveInteger	0..1	attr	CAN Identifier used to configure the CAN Transceiver for partial network wakeup.
pncWakeupCanIdExtended	Boolean	0..1	attr	Defines whether pncWakeupCanId and pncWakeupCanIdMask shall be interpreted as extended or standard CAN ID.
pncWakeupCanIdMask	PositiveInteger	0..1	attr	Bit mask for CAN Identifier used to configure the CAN Transceiver for partial network wakeup.
pncWakeupDataMask	PositiveInteger	0..1	attr	Bit mask for CAN Payload used to configure the CAN Transceiver for partial network wakeup.
pncWakeupDlc	PositiveInteger	0..1	attr	Data Length of the remote data frame used to configure the CAN Transceiver for partial network wakeup in Bytes.

Table 2.18: CanCommunicationConnector

2.3.2 TTCAN

Modeling of TTCAN clusters is supported in the System Template by the means of four specialized meta-model classes: `TtcanCluster`, `TtcanCommunicationController`, `TtcanCommunicationConnector`, `TtcanPhysicalChannel` (figure 2.5).

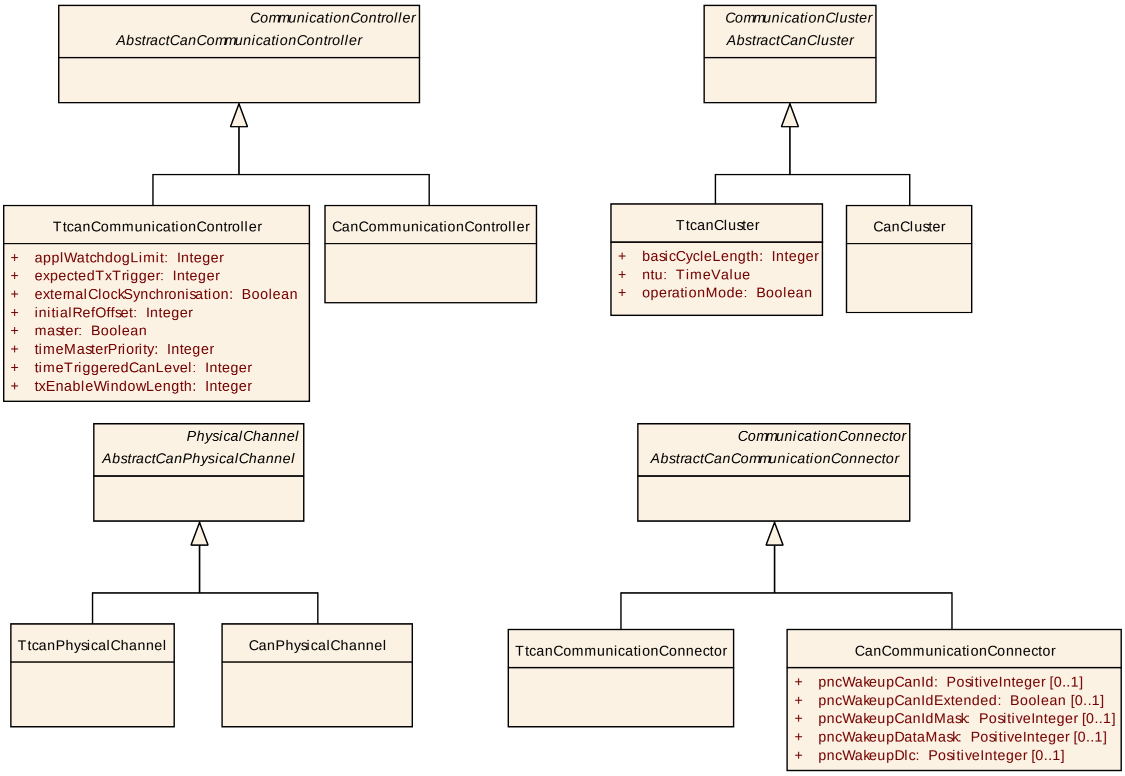

Figure 2.5: TTCAN bus elements (Fibex4Ttcan_Topology)

2.3.2.1 TTCAN Cluster

`TtcanCluster` specifies the existence of a TTCAN cluster in the system's physical topology. Additionally to the common CAN and TTCAN attributes it contains TTCAN-specific cluster-wide attributes.

Class	≪atpVariation≫ **TtcanCluster**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Ttcan::TtcanTopology			
Note	TTCAN bus specific cluster attributes. **Tags:** atp.recommendedPackage=CommunicationClusters			
Base	ARObject,AbstractCanCluster,CollectableElement,CommunicationCluster,FibexElement,Identifiable,MultilanguageReferrable,PackageableElement,Referrable			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**
basicCycleLength	Integer	1	attr	Length of a basic-cycle. Unit: NTUs
ntu	TimeValue	1	attr	Unit measuring all times and providing a constant of the whole network. For level 1, this is always the CAN bit time. Unit: seconds.
operationMode	Boolean	1	attr	Possible operation modes True: Time-Triggered False: Event-Synchronised-Time-Triggered

Table 2.19: TtcanCluster

2.3.2.2 TTCAN Communication Controller

`TtcanCommunicationController` is a specialization of the `AbstractCanCommunicationController` class. Additionally to the common CAN and TTCAN attributes it contains the specific TTCAN Controller attributes.

Class	≪atpVariation≫ **TtcanCommunicationController**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Ttcan::TtcanTopology			
Note	TTCAN bus specific communication port attributes.			
Base	ARObject,AbstractCanCommunicationController,CommunicationController,Identifiable,MultilanguageReferrable,Referrable			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**
applWatchdogLimit	Integer	1	attr	The Appl_Watchdog_Limit shall be an 8-bit value specifying the period for the application watchdog in Appl_Watchdog_Limit times 256 NTUs.
expectedTxTrigger	Integer	1	attr	The Expected_Tx_Trigger shall be an eight (8) bit value which limits the number of messages the FSE may try to transmit in one matrix cycle.
externalClockSynchronisation	Boolean	1	attr	One bit shall be used to configure whether or not external clock synchronisation will be allowed during runtime (only Level 2).
initialRefOffset	Integer	1	attr	The Initial_Ref_Offset shall be an eight (8) bit value for the initialisation of Ref_Trigger_Offset.
master	Boolean	1	attr	One bit shall be used to distinguish between (potential) time masters and time slaves. This can be derived from the frame-triggering's triggers.

Attribute	Datatype	Mul.	Kind	Note
timeMasterPriority	Integer	1	attr	The time master priority shall contain a three bit value for the priority of the current time master (the last three bits of the identifier of the reference message). This can be derived from the frame-triggering's triggers.
timeTriggeredCanLevel	Integer	1	attr	One bit shall be used to distinguish between Level 1 and Level 2.
txEnableWindowLength	Integer	1	attr	The length of the Tx_Enable window shall be a four (4) bit value specifying the length of the time period (1-16 nominal CAN bit times) in which a transmission may be started.

Table 2.20: TtcanCommunicationController

2.3.2.3 TTCAN Physical Channel

`TtcanPhysicalChannel` is a specialization of the `AbstractCanPhysicalChannel` class. Additionally to the common CAN and TTCAN attributes it contains the specific TTCAN Physical Channel attributes.

Class	**TtcanPhysicalChannel**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Ttcan::TtcanTopology			
Note	TTCAN bus specific physical channel attributes.			
Base	ARObject,AbstractCanPhysicalChannel,Identifiable,MultilanguageReferrable,PhysicalChannel,Referrable			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**
–	–	–	–	–

Table 2.21: TtcanPhysicalChannel

2.3.2.4 TTCAN Communication Connector

`TtcanCommunicationConnector` is a specialization of the `AbstractCanCommunicationConnector` class. Additionally to the common CAN and TTCAN attributes it contains the specific TTCAN `CommunicationConnector` attributes.

Class	**TtcanCommunicationConnector**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Ttcan::TtcanTopology			
Note	TTCAN bus specific communication connector attributes.			
Base	ARObject,AbstractCanCommunicationConnector,CommunicationConnector,Identifiable,MultilanguageReferrable,Referrable			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**
–	–	–	–	–

Table 2.22: TtcanCommunicationConnector

2.3.3 FlexRay

Modeling of FlexRay clusters is supported in the System Template by the means of four specialized meta-model classes: `FlexrayCluster`, `FlexrayCommunicationConnector`, `FlexrayPhysicalChannel`, `FlexrayCommunicationController` (Figure 2.6).

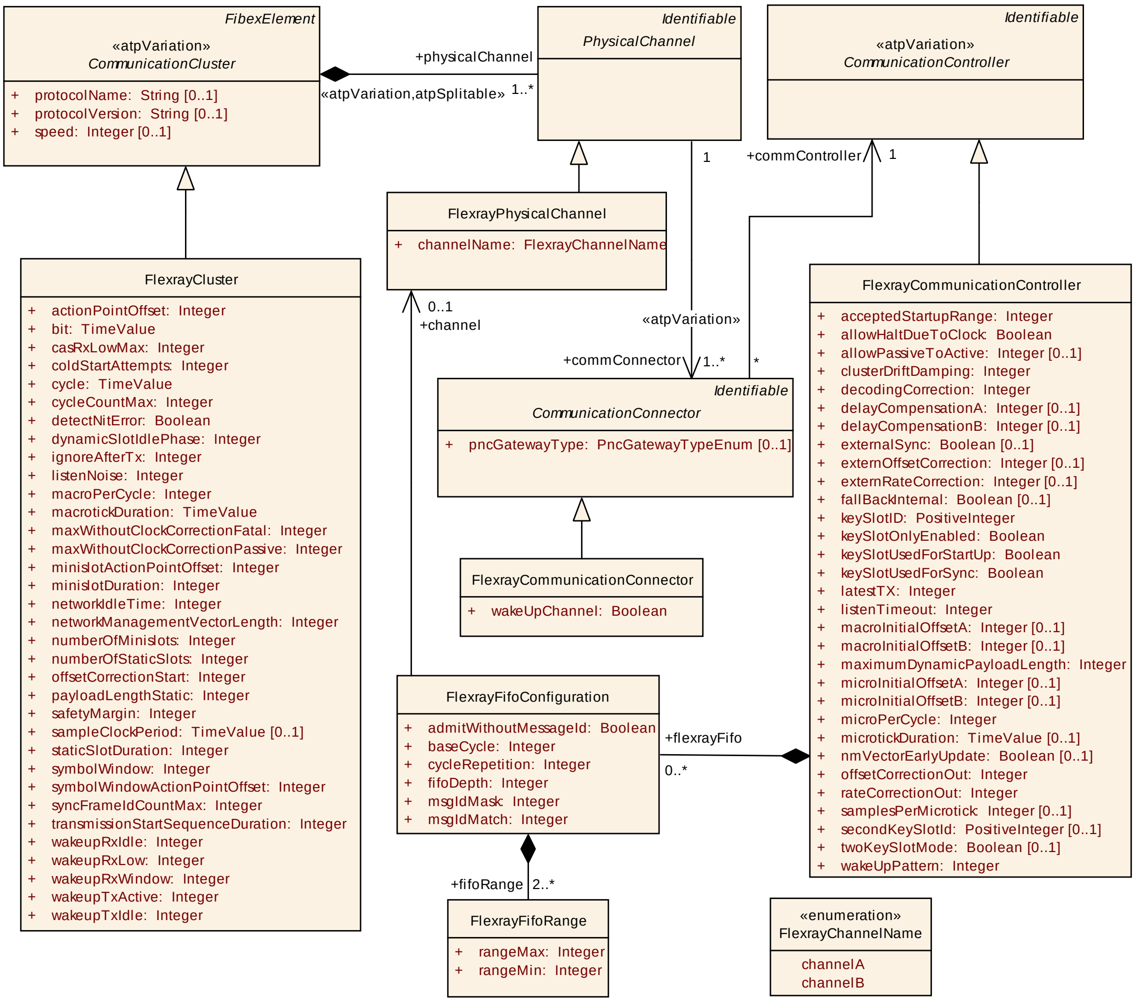

Figure 2.6: FlexRay cluster elements (Fibex4FlexRay_Topology)

2.3.3.1 FlexRay Cluster

`FlexRayCluster` specifies the existence of a FlexRay cluster in the system's physical topology. It contains additional FlexRay-specific cluster-wide attributes.

Class	≪atpVariation≫ **FlexrayCluster**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Flexray::FlexrayTopology			
Note	FlexRay specific attributes to the physicalCluster **Tags:** atp.recommendedPackage=CommunicationClusters			
Base	ARObject,CollectableElement,CommunicationCluster,Fibex Element,Identifiable,MultilanguageReferrable,PackageableElement,Referrable			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**
actionPoint Offset	Integer	1	attr	The offset of the action point in networks
bit	TimeValue	1	attr	Nominal bit time (= 1 / fx:SPEED). gdBit = cSamplesPerBit * gdSampleClockPeriod. Unit: seconds (gdBit)
casRxLow Max	Integer	1	attr	Upper limit of the Collision Avoidance Symbol (CAS) acceptance window. Unit:bitDuration
coldStartAt tempts	Integer	1	attr	The maximum number of times that a node in this cluster is permitted to attempt to start the cluster by initiating schedule synchronization
cycle	TimeValue	1	attr	Length of the cycle. Unit: seconds
cycleCount Max	Integer	1	attr	Maximum cycle counter value in a given cluster. Remark: Set to 63 for FlexRay Protocol 2.1 Rev. A compliance.
detectNitEr ror	Boolean	1	attr	Indicates whether NIT error status of each cluster shall be detected or not.
dynamicSl otIdlePhas e	Integer	1	attr	The duration of the dynamic slot idle phase in minislots.
ignoreAfter Tx	Integer	1	attr	Duration for which the bitstrobing is paused after transmission .
listenNoise	Integer	1	attr	Upper limit for the start up and wake up listen timeout in the presence of noise. Expressed as a multiple of the cluster constant pdListenTimeout. Unit microticks
macroPer Cycle	Integer	1	attr	The number of macroticks in a communication cycle
macrotickD uration	TimeValue	1	attr	Duration of the cluster wide nominal macrotick, expressed in s.
maxWithou tClockCorr ectionFatal	Integer	1	attr	Threshold concerning vClockCorrectionFailedCounter. Defines the number of consecutive even/odd Cycle pairs with missing clock correction terms that will cause the protocol to transition from the POC:normal active or POC:normal passive state into the POC:halt state.
maxWithou tClockCorr ectionPass ive	Integer	1	attr	Threshold concerning vClockCorrectionFailedCounter. Defines the number of consecutive even/odd Cycle pairs with missing clock correction terms that will cause the protocol to transition from the POC:normal active state to the POC:normal passive state.

Attribute	Datatype	Mul.	Kind	Note
minislotActionPointOffset	Integer	1	attr	The Offset of the action point within a minislot. Unit: macroticks
minislotDuration	Integer	1	attr	The duration of a minislot (dynamic segment). Unit: macroticks.
networkIdleTime	Integer	1	attr	The duration of the network idle time in macroticks
networkManagementVectorLength	Integer	1	attr	Length of the Network Management vector in a cluster
numberOfMinislots	Integer	1	attr	Number of Minislots in the dynamic segment.
numberOfStaticSlots	Integer	1	attr	The number of static slots in the static segment.
offsetCorrectionStart	Integer	1	attr	Start of the offset correction phase within the Network Idle Time (NIT), expressed as the number of macroticks from the start of cycle. Unit: macroticks
payloadLengthStatic	Integer	1	attr	Globally configured payload length of a static frame. Unit: 16-bit WORDS.
safetyMargin	Integer	1	attr	Additional timespan in macroticks which takes jitter into account to be able to set the JobListPointer to the next possible job which can be executed in case the FlexRay Job List Execution Function has be resynchronized.
sampleClockPeriod	TimeValue	0..1	attr	Sample clock period. Unit: seconds
staticSlotDuration	Integer	1	attr	The duration of a slot in the static segment. Unit: macroticks
symbolWindow	Integer	1	attr	The duration of the symbol window. Unit: macroticks
symbolWindowActionPointOffset	Integer	1	attr	Number of macroticks the action point offset is from the beginning of the symbol window .
syncFrameIdCountMax	Integer	1	attr	Maximum number of distinct syncframe identifiers present in a given cluster. This parameter maps to FlexRay Protocol 2.1 Rev. A parameter gSyncNodeMax.
transmissionStartSequenceDuration	Integer	1	attr	Number of bits in the Transmission Start Sequence .
wakeupRxIdle	Integer	1	attr	Number of bits used by the node to test the duration of the 'idle' or HIGH phase of a received wakeup. Unit:bitDuration Remarks: This parameter maps to FlexRay Protocol 2.1 Rev. A parameter gdWakeupSymbolRxIdle.

Attribute	Datatype	Mul.	Kind	Note
wakeupRx Low	Integer	1	attr	Number of bits used by the node to test the duration of the LOW phase of a received wakeup. Unit:bitDuration Remarks: This parameter maps to FlexRay Protocol 2.1 Rev. A parameter gdWakeupSymbolRxLow.
wakeupRx Window	Integer	1	attr	The size of the window used to detect wakeups . Remarks: This parameter maps to FlexRay Protocol 2.1 Rev. A parameter gdWakeupSymbolRxWindow.
wakeupTx Active	Integer	1	attr	Number of bits used by the node to transmit the LOW phase of awakeup symbol and the HIGH and LOW phases of a WUDOP. Unit:bitDuration
wakeupTxI dle	Integer	1	attr	Number of bits used by the node to transmit the 'idle' part of a wakeup symbol. Unit: gDbit

Table 2.23: FlexrayCluster

2.3.3.2 FlexRay Communication Controller

`FlexrayCommunicationController` is a specialization of the `Communication-Controller` class. It contains the specific FlexRay controller attributes needed for configuring the FlexRay stack in an ECU connected to a certain FlexRay cluster.

Class	≪`atpVariation`≫ **FlexrayCommunicationController**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Flexray::FlexrayTopology			
Note	FlexRay bus specific communication port attributes.			
Base	ARObject,CommunicationController,Identifiable,MultilanguageReferrable,Referrable			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**
acceptedSt artupRang e	Integer	1	attr	Expanded range of measured clock deviation allowed for startup frames during integration. Unit:microtick
allowHaltD ueToClock	Boolean	1	attr	Boolean flag that controls the transition to the POC:halt state due to a clock synchronization errors. If set to true, the Communication Controller is allowed to transition to POC:halt. If set to false, the Communication Controller will not transition to the POC:halt state but will enter or remain in the normal POC (passive State).
allowPassi veToActive	Integer	0..1	attr	Number of consecutive even/odd cycle pairs that must have valid clock correction terms before the Communication Controller will be allowed to transition from the POC:normal passive state to POC:normal active state. If set to 0, the Communication Controller is not allowed to transition from POC:norm
clusterDrift Damping	Integer	1	attr	The cluster drift damping factor used in clock synchronization rate correction in microticks

Attribute	Datatype	Mul.	Kind	Note
decodingCorrection	Integer	1	attr	Value used by the receiver to calculate the difference between primary time reference point and secondary time reference point. Unit: Microticks (pDecodingCorrection)
delayCompensationA	Integer	0..1	attr	Value used to compensate for reception delays on channel A Unit: Microticks. This optional parameter shall only be filled out if channel A is used.
delayCompensationB	Integer	0..1	attr	Value used to compensate for reception delays on channel B. Unit: Microticks. This optional parameter shall only be filled out if channel B is used.
externOffsetCorrection	Integer	0..1	attr	Fixed amount added or subtracted to the calculated offset correction term to facilitate external offset correction, expressed in node-local microticks.
externRateCorrection	Integer	0..1	attr	Fixed amount added or subtracted to the calculated rate correction term to facilitate external rate correction, expressed in node-local microticks.
externalSync	Boolean	0..1	attr	Flag indicating whether the node is externally synchronized (operating as Time Gateway Sink in an TT-E Time Triggered External Sync cluster) or locally synchronized.
fallBackInternal	Boolean	0..1	attr	Flag indicating whether a Time Gateway Sink node will switch to local clock operation when synchronization with the Time Gateway Source node is lost (pFallBackInternal = true) or will instead go to POC:ready (pFallBackInternal = false).
flexrayFifo	FlexrayFifoConfiguration	*	aggr	One First In First Out (FIFO) queued receive structure, defining the admittance criteria to the FIFO.
keySlotID	PositiveInteger	1	attr	ID of the slot used to transmit the startup frame, sync frame, or designated single slot frame.
keySlotOnlyEnabled	Boolean	1	attr	Flag indicating whether or not the node shall enter key slot only mode following startup.
keySlotUsedForStartUp	Boolean	1	attr	Flag indicating whether the Key Slot is used to transmit a startup frame.
keySlotUsedForSync	Boolean	1	attr	Flag indicating whether the Key Slot is used to transmit a sync frame.
latestTX	Integer	1	attr	The number of the last minislot in which a transmission can start in the dynamic segment for the respective node
listenTimeout	Integer	1	attr	Value for the startup listen timeout and wakeup listen timeout. Although this is a node local parameter, the real time equivalent of this value should be the same for all nodes in the cluster. Unit: Microticks

Attribute	Datatype	Mul.	Kind	Note
macroInitialOffsetA	Integer	0..1	attr	Integer number of macroticks between the static slot boundary and the closest macrotick boundary of the secondary time reference point based on the nominal macrotick duration. (pMacroInitialOffset). This optional parameter shall only be filled out if channel A is used.
macroInitialOffsetB	Integer	0..1	attr	Integer number of macroticks between the static slot boundary and the closest macrotick boundary of the secondary time reference point based on the nominal macrotick duration. (pMacroInitialOffset). This optional parameter shall only be filled out if channel B is used.
maximumDynamicPayloadLength	Integer	1	attr	Maximum payload length for the dynamic channel of a frame in 16 bit WORDS.
microInitialOffsetA	Integer	0..1	attr	Number of microticks between the closest macrotick boundary described by gMacroInitialOffset and the secondary time reference point. The parameter depends on pDelayCompensationA and therefore it has to be set independently for each channel. This optional parameter shall only be filled out if channel A is used.
microInitialOffsetB	Integer	0..1	attr	Number of microticks between the closest macrotick boundary described by gMacroInitialOffset and the secondary time reference point. The parameter depends on pDelayCompensationB and therefore it has to be set independently for each channel. This optional parameter shall only be filled out if channel B is used.
microPerCycle	Integer	1	attr	The nominal number of microticks in a communication cycle
microtickDuration	TimeValue	0..1	attr	Duration of a microtick. This attribute can be derived from samplePerMicrotick and gdSampleClockPeriod. Unit: seconds
nmVectorEarlyUpdate	Boolean	0..1	attr	Flag indicating when the update of the Network Management Vector in the CHI shall take place. If set to false, the update shall take place after the NIT. If set to true, the update shall take place after the end of the static segment.
offsetCorrectionOut	Integer	1	attr	Magnitude of the maximum permissible offset correction value. Unit:microtick (pOffsetCorrectionOut)

Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
rateCorrectionOut	Integer	1	attr	Magnitude of the maximum permissible rate correction value and the maximum drift offset between two nodes operating with unsynchronized clocks for one communication cycle. Unit:Microticks (pRateCorrectionOut) Remarks: This parameter maps to FlexRay Protocol 2.1 Rev. A parameter pdMaxDrift.
samplesPerMicrotick	Integer	0..1	attr	Number of samples per microtick
secondKeySlotId	PositiveInteger	0..1	attr	ID of the second Key slot, in which a second startup frame shall be sent in TT-L Time Triggered Local Master Sync or TT-E Time Triggered External Sync mode. If this parameter is set to zero the node does not have a second key slot.
twoKeySlotMode	Boolean	0..1	attr	Flag indicating whether node operates as a startup node in a TT-E Time Triggered External Sync or TT-L Time Triggered Local Master Sync cluster.
wakeUpPattern	Integer	1	attr	Number of repetitions of the Tx-wakeup symbol to be sent during the CC_WakeupSend state of this Node in the cluster

Table 2.24: FlexrayCommunicationController

Class	**FlexrayFifoConfiguration**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Flexray::FlexrayTopology			
Note	One First In First Out (FIFO) queued receive structure, defining the admittance criteria to the FIFO, and mandating the ability to admit messages into the FIFO based on Message Id filtering criteria.			
Base	ARObject			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**
admitWithoutMessageId	Boolean	1	attr	Boolean configuration which determines whether or not frames received in the dynamic segment that don't contain a message ID will be admitted into the FIFO.
baseCycle	Integer	1	attr	FIFO cycle counter acceptance criteria.
channel	FlexrayPhysicalChannel	0..1	ref	Fifo channel admittance criteria.
cycleRepetition	Integer	1	attr	FIFO cycle counter acceptance criteria.
fifoDepth	Integer	1	attr	Fifo Depth.
fifoRange	FlexrayFifoRange	2..*	aggr	FIFO Frame Id range acceptance criteria.
msgIdMask	Integer	1	attr	FIFO message identifier acceptance criteria (Mask filter).
msgIdMatch	Integer	1	attr	FIFO message identifier acceptance criteria (Match filter).

Table 2.25: FlexrayFifoConfiguration

Class	**FlexrayFifoRange**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Flexray::FlexrayTopology			
Note	FIFO Frame Id range acceptance criteria.			
Base	ARObject			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**
rangeMax	Integer	1	attr	Max Range.
rangeMin	Integer	1	attr	Min Range.

Table 2.26: FlexrayFifoRange

2.3.3.3 FlexRay Communication Connector

FlexrayCommunicationConnector adds the FlexRay specific attributes to the CommunicationConnector.

Class	**FlexrayCommunicationConnector**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Flexray::FlexrayTopology			
Note	FlexRay specific attributes to the CommunicationConnector			
Base	ARObject,CommunicationConnector,Identifiable,MultilanguageReferrable,Referrable			
Attribute	*Datatype*	*Mul.*	*Kind*	*Note*
wakeUpChannel	Boolean	1	attr	Referenced channel used by the node to send a wakeup pattern. (pWakeupChannel)

Table 2.27: FlexrayCommunicationConnector

2.3.3.4 FlexRay Physical Channel

FlexrayPhysicalChannel adds the FlexRay specific attributes to the PhysicalChannel.

Class	**FlexrayPhysicalChannel**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Flexray::FlexrayTopology			
Note	FlexRay specific attributes to the physicalChannel			
Base	ARObject,Identifiable,MultilanguageReferrable,PhysicalChannel,Referrable			
Attribute	*Datatype*	*Mul.*	*Kind*	*Note*
channelName	FlexrayChannelName	1	attr	Name of the channel (Channel A or Channel B).

Table 2.28: FlexrayPhysicalChannel

— AUTOSAR CONFIDENTIAL —

Enumeration	**FlexrayChannelName**
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Flexray::Flexray Topology
Note	Name of the channel.
Literal	*Description*
channelA	Channel A
channelB	Channel B

Table 2.29: FlexrayChannelName

[constr_3018] Number of FlexRay channels ⌈A FlexRay cluster shall use either one channel with `FlexrayChannelName` Channel A or two channels with one `FlexrayChannelName` Channel A and one `FlexrayChannelName` Channel B. ⌋

2.3.4 LIN

A LinCluster consists of exactly one master node connected to several slave nodes. The master is responsible for providing the frame headers on the bus according to a predefined schedule, whereas the slaves send or receive the actual frame information ([8]).

In the System Template the different properties of master and slave nodes are handled by deriving the LIN-specific subclasses LinMaster and LinSlave as specializations of LINCommunicationController.

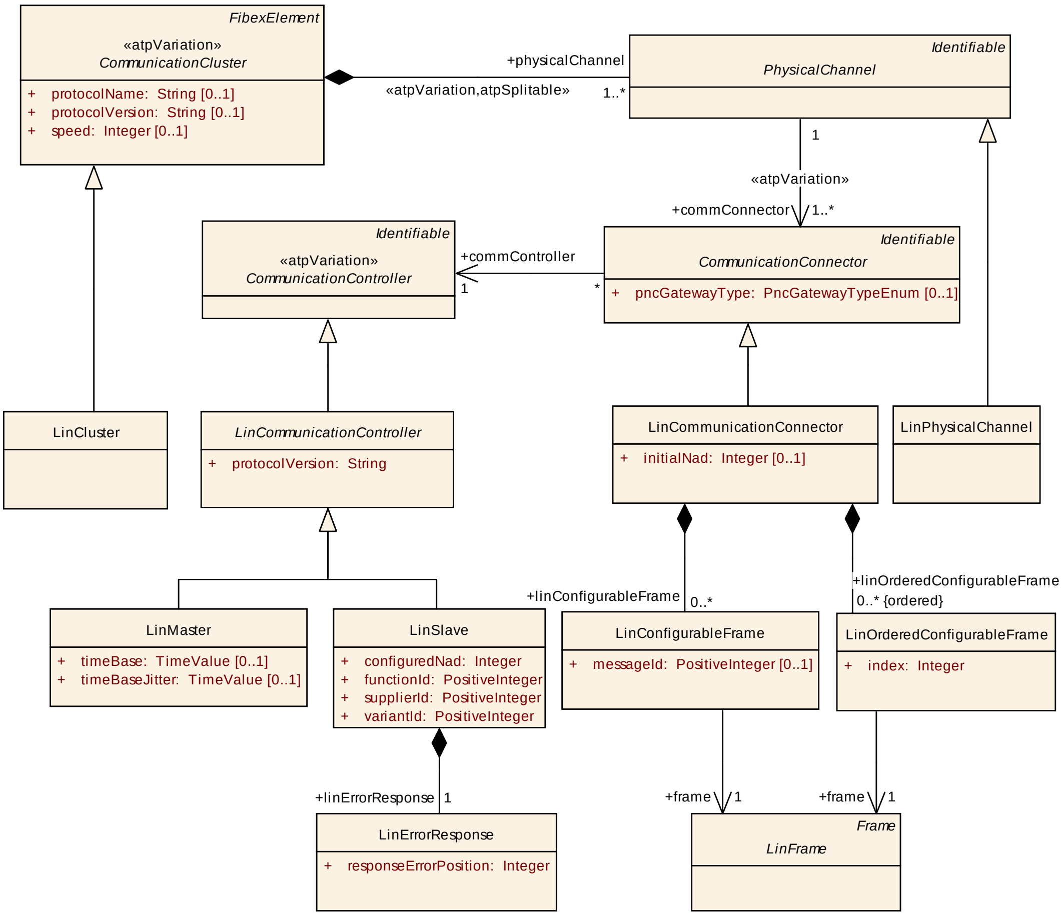

Figure 2.7: Specialized `LINCommunicationController` attributes (Fibex4Lin_Topology)

Note that the AUTOSAR BSW only supports LIN masters. LIN slaves are seen as non AUTOSAR ECUs. They can be described in the System Template in order to configure the LIN Interface for the master correctly, but AUTOSAR does not support the development of LIN slaves as of AUTOSAR release 4.0 ([13], [14]).

2.3.4.1 LIN Cluster

LinCluster specifies the existence of a LIN cluster in the system's physical topology.

Class	≪atpVariation≫ **LinCluster**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Lin::LinTopology			
Note	LIN specific attributes **Tags:** atp.recommendedPackage=CommunicationClusters			
Base	ARObject,CollectableElement,CommunicationCluster,Fibex Element,Identifiable,MultilanguageReferrable,PackageableElement,Referrable			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**
–	–	–	–	–

Table 2.30: LinCluster

2.3.4.2 LIN Communication Controller

LINCommunicationController is a specialization of the CommunicationController class. It is an abstract class, to be further specialized by LinMaster and LinSlave.

Class	≪atpVariation≫ **LinCommunicationController (abstract)**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Lin::LinTopology			
Note	LIN bus specific communication controller attributes.			
Base	ARObject,CommunicationController,Identifiable,MultilanguageReferrable,Referrable			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**
protocolVersion	String	1	attr	Version specifier for a communication protocol.

Table 2.31: LinCommunicationController

2.3.4.3 LIN Master

LinMaster describes the existence of a LIN master task in a LIN topology node. As such it contains the attributes specific to a LIN master task.

Class	≪atpVariation≫ **LinMaster**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Lin::LinTopology			
Note	Describing the properties of the refering ecu as a LIN master.			
Base	ARObject,CommunicationController,Identifiable,LinCommunication Controller,MultilanguageReferrable,Referrable			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**

— AUTOSAR CONFIDENTIAL —

Attribute	Datatype	Mul.	Kind	Note
timeBase	TimeValue	0..1	attr	Time base is mandatory for the master. It is not used for slaves. LIN 2.0 Spec states: "The time_base value specifies the used time base in the master node to generate the maximum allowed frame transfer time." The time base shall be specified AUTOSAR conform in seconds.
timeBaseJitter	TimeValue	0..1	attr	The attribute timeBaseJitter is a mandatory attribute for the master and not used for slaves. LIN 2.0 Spec states: "The jitter value specifies the differences between the maximum and minimum delay from time base start point to the frame header sending start point (falling edge of BREAK signal)." The jitter shall be specified AUTOSAR conform in seconds.

Table 2.32: LinMaster

2.3.4.4 LIN Slave

`LinSlave` describes the existence of a LIN slave task in a LIN topology node. It describes the attributes of a single LIN slave node. AUTOSAR doesn't support LIN slave functionality in an AUTOSAR ECU, thus not the full FIBEX description of a slave node, but rather the subset of attributes of a Node Capability File (ncf, see [8]) relevant as requirements for configuring the master are included in the System Template.

Class	≪**atpVariation**≫ **LinSlave**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Lin::LinTopology			
Note	Describing the properties of the referring ecu as a LIN slave.			
Base	ARObject,CommunicationController,Identifiable,LinCommunicationController,MultilanguageReferrable,Referrable			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
configuredNad	Integer	1	attr	To distinguish LIN slaves that are used twice or more within the same cluster.
functionId	PositiveInteger	1	attr	LIN function ID
linErrorResponse	LinErrorResponse	1	aggr	Each slave node shall publish one response error in one of its transmitted unconditional frames.
supplierId	PositiveInteger	1	attr	LIN Supplier ID
variantId	PositiveInteger	1	attr	Specifies the Variant ID

Table 2.33: LinSlave

Class	LinErrorResponse			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Lin::LinCommunication			
Note	Each slave node shall publish a one bit signal, named response_error, to the master node in one of its transmitted unconditional frames. The response_error signal shall be set whenever a frame (except for event triggered frame responses) that is transmitted or received by the slave node contains an error in the frame response. The response_error signal shall be cleared when the unconditional frame containing the response_error signal is successfully transmitted.			
Base	ARObject			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**
frameTriggering	LinFrameTriggering	1	ref	Reference to an unconditional frame that transmits the response error. The referenced LinFrameTriggering shall contain a reference to an unconditionalFrame.
responseErrorPosition	Integer	1	attr	Specifies the position of the ResponseError bit in the frame. Each slave node shall publish one response error in one of its transmitted unconditional frames.

Table 2.34: LinErrorResponse

2.3.4.5 LIN Communication Connector

`LINCommunicationConnector` is a specialization of the `CommunicationConnector` class. The `LINCommunicationConnector` element contains lists of frames processed by the slave node. For the LIN 2.0 Assign-Frame command the `LinConfigurableFrame` list shall be used. For the LIN 2.1 Assign-Frame-PID-Range command the `LinOrderedConfigurableFrame` list shall be used.

Class	LinCommunicationConnector			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Lin::LinTopology			
Note	LIN bus specific communication connector attributes.			
Base	ARObject,CommunicationConnector,Identifiable,MultilanguageReferrable,Referrable			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**
initialNad	Integer	0..1	attr	Initial NAD of the LIN slave.
linConfigurableFrame	LinConfigurableFrame	*	aggr	LinConfigurableFrames shall list all frames (unconditional frames, event-triggered frames and sporadic frames) processed by the slave node. This element is necessary for the LIN 2.0 Assign-Frame command.
linOrderedConfigurableFrame (ordered)	LinOrderedConfigurableFrame	*	aggr	LinOrderedConfigurableFrames shall list all frames (unconditional frames, event-triggered frames and sporadic frames) processed by the slave node. This element is necessary for the LIN 2.1 Assign-Frame-PID-Range command.

Table 2.35: LinCommunicationConnector

Class	LinConfigurableFrame			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Lin::LinTopology			
Note	Assignment of messageIds to Frames. This element shall be used for the LIN 2.0 Assign-Frame command.			
Base	ARObject			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**
frame	LinFrame	1	ref	Reference to a Frame that is processed by the slave node.
messageId	PositiveInteger	0..1	attr	MessageId for the referenced frame

Table 2.36: LinConfigurableFrame

Class	LinOrderedConfigurableFrame			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Lin::LinTopology			
Note	With the assignment of the index to a frame a mapping of Pids to Frames is possible. This element shall be used for the LIN 2.1 Assign-Frame-PID-Range command.			
Base	ARObject			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**
frame	LinFrame	1	ref	Reference to a Frame that is processed by the slave node.
index	Integer	1	attr	This attribute is used to order the elements and allows an assignment of Pids to ConfigurableFrames that are defined in the slave.

Table 2.37: LinOrderedConfigurableFrame

2.3.4.6 LIN Physical Channel

`LinPhysicalChannel` is a specialization of the `PhysicalChannel` class. It contains additional Lin-specific `PhysicalChannel` attributes.

Class	LinPhysicalChannel			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Lin::LinTopology			
Note	LIN specific attributes to the physicalChannel			
Base	ARObject,Identifiable,MultilanguageReferrable,PhysicalChannel,Referrable			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**
scheduleTable	LinScheduleTable	*	aggr	Schedule tables organize the timings of the frames for LIN. atpVariation: If the transmitted frames are variable, the corresponding ScheduleTables must be variable, too. **Stereotypes:** atpVariation **Tags:** Vh.latestBindingTime=PostBuild

Table 2.38: LinPhysicalChannel

[constr_3015] Number of LIN channels ⌈LIN clusters shall aggregate exactly one PhysicalChannel.⌋

2.3.5 Ethernet

Modeling of the Ethernet bus is supported in the System Template by the means of four specialized meta-model classes: `EthernetCluster`, `EthernetCommunicationController`, `EthernetCommunicationConnector`, `EthernetPhysicalChannel`. (Figure 2.8).

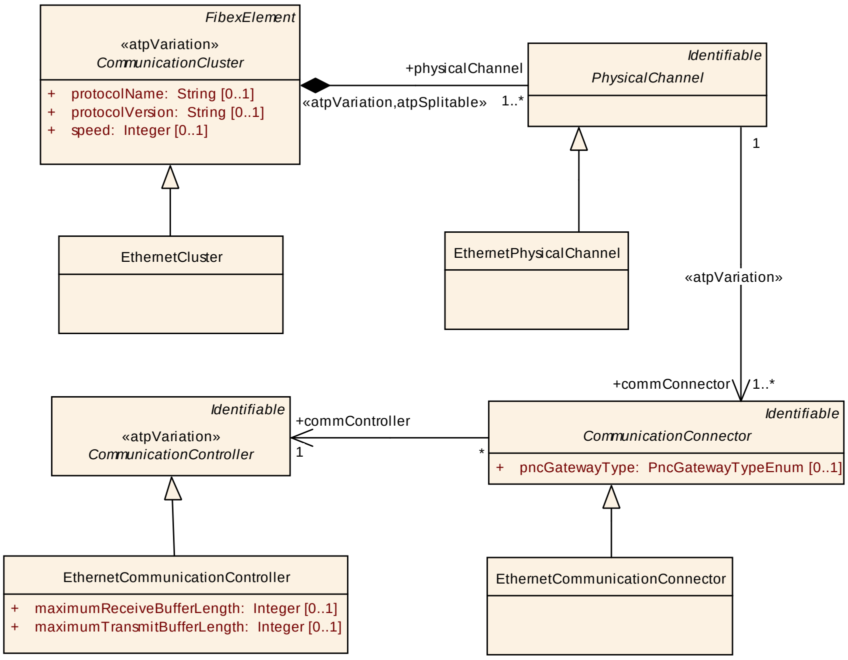

Figure 2.8: Ethernet topology elements (Fibex4Ethernet_Topology)

2.3.5.1 Ethernet Cluster

`EthernetCluster` specifies the existence of a Ethernet cluster in the system's physical topology. It contains additional Ethernet-specific, cluster-wide attributes.

Class	≪**atpVariation**≫ **EthernetCluster**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Ethernet::Ethernet Topology			
Note	Ethernet specific cluster attributes. **Tags:** atp.recommendedPackage=CommunicationClusters			
Base	ARObject,CollectableElement,CommunicationCluster,Fibex Element,Identifiable,MultilanguageReferrable,PackageableElement,Referrable			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
–	–	–	–	–

Table 2.39: EthernetCluster

2.3.5.2 Ethernet Communication Controller

`EthernetCommunicationController` is a specialization of the `Communication-Controller` class. It contains the specific Ethernet controller attributes needed for configuring an ECU connected to a certain Ethernet cluster.

Class	≪atpVariation≫ **EthernetCommunicationController**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Ethernet::Ethernet Topology			
Note	Ethernet specific communication port attributes.			
Base	ARObject,CommunicationController,Identifiable,MultilanguageReferrable,Referrable			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
maximumReceiveBufferLength	Integer	0..1	attr	Determines the maximum receive buffer length (frame length) in bytes.
maximumTransmitBufferLength	Integer	0..1	attr	Determines the maximum transmit buffer length (frame length) in bytes.

Table 2.40: EthernetCommunicationController

2.3.5.3 Ethernet Communication Connector

`EthernetCommunicationConnector` adds the Ethernet specific attributes to the `CommunicationConnector`.

Class	**EthernetCommunicationConnector**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Ethernet::Ethernet Topology			
Note	Ethernet specific attributes to the CommunicationConnector			
Base	ARObject,CommunicationConnector,Identifiable,MultilanguageReferrable,Referrable			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
–	–	–	–	–

Table 2.41: EthernetCommunicationConnector

— AUTOSAR CONFIDENTIAL —

2.3.5.4 Ethernet Physical Channel

EthernetPhysicalChannel adds the Ethernet specific attributes to the PhysicalChannel.

Class	**EthernetPhysicalChannel**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Ethernet::Ethernet Topology			
Note	Ethernet specific attributes to the PhysicalChannel.			
Base	ARObject,Identifiable,MultilanguageReferrable,PhysicalChannel,Referrable			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**
soAdConfig	SoAdConfig	1	aggr	SoAd Configuration for one specific Physical Channel.

Table 2.42: EthernetPhysicalChannel

[constr_3016] Number of Ethernet channels ⌈Ethernet clusters shall aggregate exactly one PhysicalChannel. ⌋

2.4 Mapping of Topology Entities onto Hardware Elements

As explained in the previous sections, the System Template contains all classes necessary to describe the physical topology in an AUTOSAR system. Based on this description, the communication matrix can be realized as explained in chapter 5.

Additionally, it is possible to map the hardware related topology elements onto their counterpart definitions in the ECU Resource Template (Figure 2.9). It can be specified which HwElement is realizing each given ECUInstance, providing the means for algorithms to map software components onto the systems ECUInstance. By specifying which hwCommunicationPort[1] on a hwCommunicationController[2] implements the topologie's CommunicationConnector on a CommunicationController, the hardware-oriented parameters in the Communication-drivers may be derived in ECU configuration phase.

Please note that this is a rather specific type of mapping, optionally binding ECU-local topology elements to specific hardware resources. It should not be confused with the System Mapping part of the System Description, where system-wide mapping decisions are described, like e.g. the the mapping of Software Components onto ECUs or the mapping of Data Element Prototypes onto System Signals (for the System Mapping, see chapter 4).

[1] HwPinGroup which is of category Communication Port
[2] HwElement which is of category Communication Controller

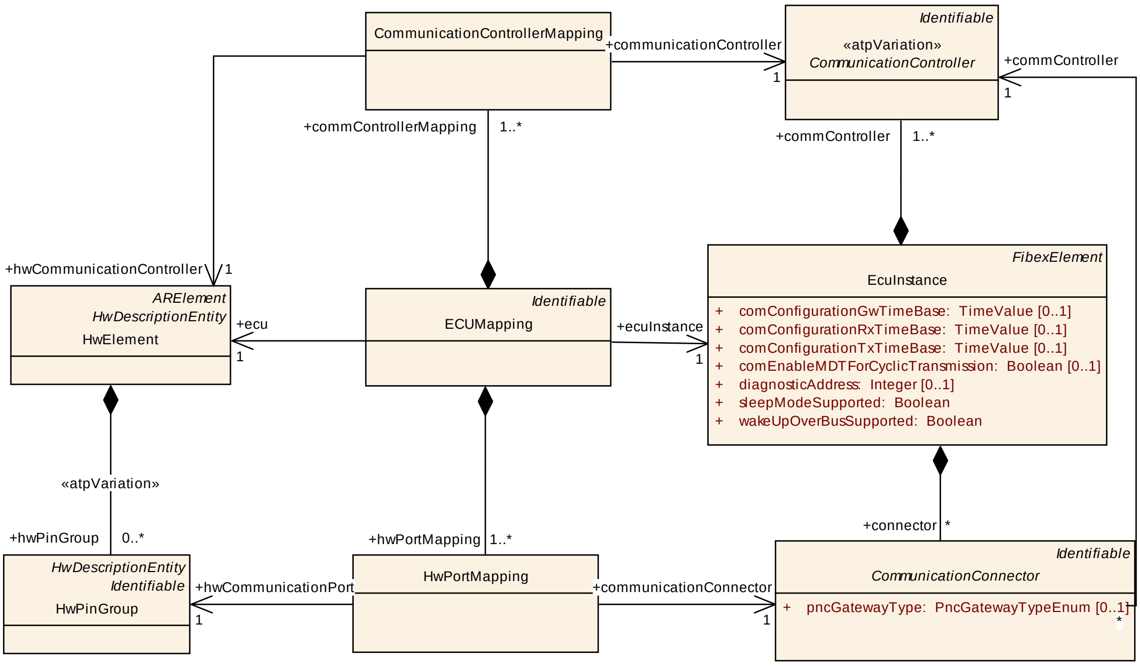

Figure 2.9: Mapping of topology description elements in the System Template onto hardware elements defined in the ECU Resource Template (ECUResourceMapping)

[constr_3006] valid EcuMapping ⌈The referenced `hwCommunicationController` and `hwCommunicationPort` shall be part of the referenced Ecu.

ECUMapping.ecu.nestedElement contains ECUMapping.communicationControllerMapping.hwCommunicationController

ECUMapping.ecu.nestedElement contains ECUMapping.hwPortMapping.hwCommunicationPort ⌋

2.4.1 ECU Mapping

`ECUMapping` allows to assign a `HwElement` to an `ECUInstance` used in a physical topology. A `HwElement` of category `ECU` is defined in the ECU Resource Template; it provides information about the internal hardware structure of an ECU. This information can be used by the System Generator to assign or validate the mapping of Software Component Prototypes onto `ECUInstances`.

An `ECUInstance` can be defined in a stand alone and reusable way. If an `ECUInstance` is assigned to a `HwElement` it shall be made clear that the `ECUInstance` actually belongs to the same System as the `ECUMapping`.

Class	ECUMapping			
Package	M2::AUTOSARTemplates::SystemTemplate::ECUResourceMapping			
Note	ECUMapping allows to assign an ECU hardware type (defined in the ECU Resource Template) to an ECUInstance used in a physical topology.			
Base	ARObject,Identifiable,MultilanguageReferrable,Referrable			
Attribute	Datatype	Mul.	Kind	Note
commControllerMapping	CommunicationControllerMapping	1..*	aggr	The ECUMapping contains the mapping of all CommunicationControllers of the ECU.
ecu	HwElement	1	ref	Reference to the Ecu description in the ECU Resource Template.
ecuInstance	EcuInstance	1	ref	Reference to the EcuInstance in the System Template
hwPortMapping	HwPortMapping	1..*	aggr	The ECUMapping contains the mapping of all HW Communication Ports of the ECU.

Table 2.43: ECUMapping

2.4.2 Communication Controller Mapping

`CommunicationControllerMapping` specifies the `HwElement` to realize the specified `CommunicationController` in a physical topology. The information may e.g. be used during ECU configuration for configuring the hardware related parameters in the communication drivers.

Class	CommunicationControllerMapping			
Package	M2::AUTOSARTemplates::SystemTemplate::ECUResourceMapping			
Note	CommunicationControllerMapping specifies the CommunicationPeripheral hardware (defined in the ECU Resource Template) to realize the specified CommunicationController in a physical topology.			
Base	ARObject			
Attribute	Datatype	Mul.	Kind	Note
communicationController	CommunicationController	1	ref	Reference to the CommunicationController in the System Template
hwCommunicationController	HwElement	1	ref	Reference to the hwCommunicationController in the ECU Resource Template.

Table 2.44: CommunicationControllerMapping

2.4.3 HW-Port Mapping

`HWPortMapping` specifies the hardware to realize the specified `Communication-Connector` in a physical topology. The information may e.g. be used during ECU configuration for configuring the hardware related parameters in the communication drivers.

Class	**HwPortMapping**			
Package	M2::AUTOSARTemplates::SystemTemplate::ECUResourceMapping			
Note	HWPortMapping specifies the hwCommunicationPort (defined in the ECU Resource Template) to realize the specified CommunicationConnector in a physical topology.			
Base	ARObject			
Attribute	*Datatype*	*Mul.*	*Kind*	*Note*
communicationConnector	CommunicationConnector	1	ref	Reference to the CommunicationConnector in the System Template
hwCommunicationPort	HwPinGroup	1	ref	Reference to the HwPinPortGroup of category CommunicationPort. The connection to the HwCommunicationController is described in the Ecu Resource Description.

Table 2.45: HwPortMapping

3 Top-level Software Composition

One of the most important inputs for the System Generator is the knowledge about the Application Software Components, their communications capabilities and the connections between them: Each SystemSignal (chapter 5.3) that is going to be exchanged between mapped Software Components onto different ECUs is a consequence of a connection between such application Software Components.

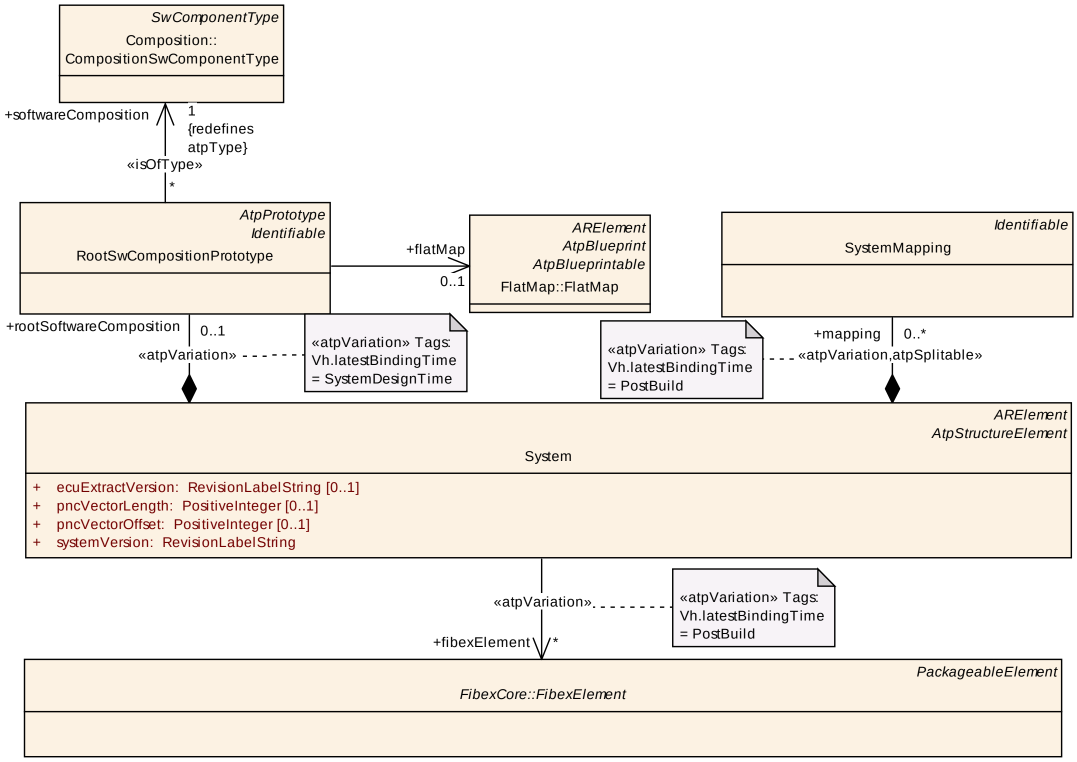

Figure 3.1: Inclusion of a (top-level) Software Composition into an AUTOSAR system (SystemTemplate)

In AUTOSAR, Software Components can either be atomic (AtomicSwComponentType) or may consist of a composition of other Software Components CompositionSwComponentType [4]. In order to assemble non-trivial applications from AUTOSAR components, such compositions can be built up hierarchically, until the outermost CompositionSwComponentType forms a kind of top-level composition.

In a complete System Description this outermost composition has the unique feature that it doesn't have any outside ports, but all the SWC contained in it are connected to each other and fully specified by their ComponentTypes, PortPrototypes, Port-Interfaces, DataElementPrototypes, InternalBehavior etc. In an System- and Ecu Extract outside ports for the outermost composition are allowed. Since the System/Ecu Extract represents the view on one Ecu, there may be the need to define the communication of this extract with the outside world.

— AUTOSAR CONFIDENTIAL —

Two approaches are available how the external communication of an ECU in the System Extract is described. In section 8.2 the communication mapping is performed in the hierarchical structure of software components. In section 8.3 external communication delegation ports are added to the System extract outermost composition. Each delegated port is connected via a `DelegationConnector` with ports of the included components that are used for the external communication.

A `System` considers such a top-level `CompositionSwComponentType` as its application software system input by owning exactly one `RootSwCompositionPrototype` class, which points to the `CompositionSwComponentType` forming the input via its ≪`isOfType`≫ relationship as shown in Figure 3.1.

By using composition, an AUTOSAR `System` uses the specialized prototype class `RootSwCompositionPrototype` in order to designate the referenced `CompositionSwComponentType` as the top-level software composition.

Class	**RootSwCompositionPrototype**			
Package	M2::AUTOSARTemplates::SystemTemplate			
Note	The RootSwCompositionPrototype represents the top-level-compostion of software components within a given System. According to the use case of the System, this may for example be the 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 PortPrototypes, PortInterfaces, VariableDataPrototypes, SwcInternalBehavior etc.), and their ports are interconnected using SwConnectorPrototypes.			
Base	ARObject,AtpFeature,AtpPrototype,Identifiable,MultilanguageReferrable,Referrable			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
calibrationParameterValueSet	CalibrationParameterValueSet	*	ref	Used CalibrationParameterValueSet for instance specific initialization of calibration parameters.
flatMap	FlatMap	0..1	ref	The FlatMap used in the scope of this RootSwCompositionPrototype.
softwareComposition	CompositionSwComponentType	1	tref	We assume that there is exactly one top-level composition that includes all Component instances of the system **Stereotypes:** isOfType

Table 3.1: RootSwCompositionPrototype

4 Mapping

A central part of the system generation process is the mapping of software components (SwComponentPrototypes) to ECUs, and the subsequent mapping of the communication between these software components to bus frames. Input to the software component mapping is the RootSwCompositionPrototype, which describes which software components have to be mapped, and the System Topology, which defines the ECU instances that are available as mapping targets. Once this mapping is done, also the communication matrix has to be taken into account for the next mapping step, the mapping of data elements exchanged between software components to bus frames. This communication matrix may either be predefined, or may be generated as part of this second mapping step. In the metamodel, different aspects of these mapping are aggregated by the meta class SystemMapping, as shown in Figure 4.1.

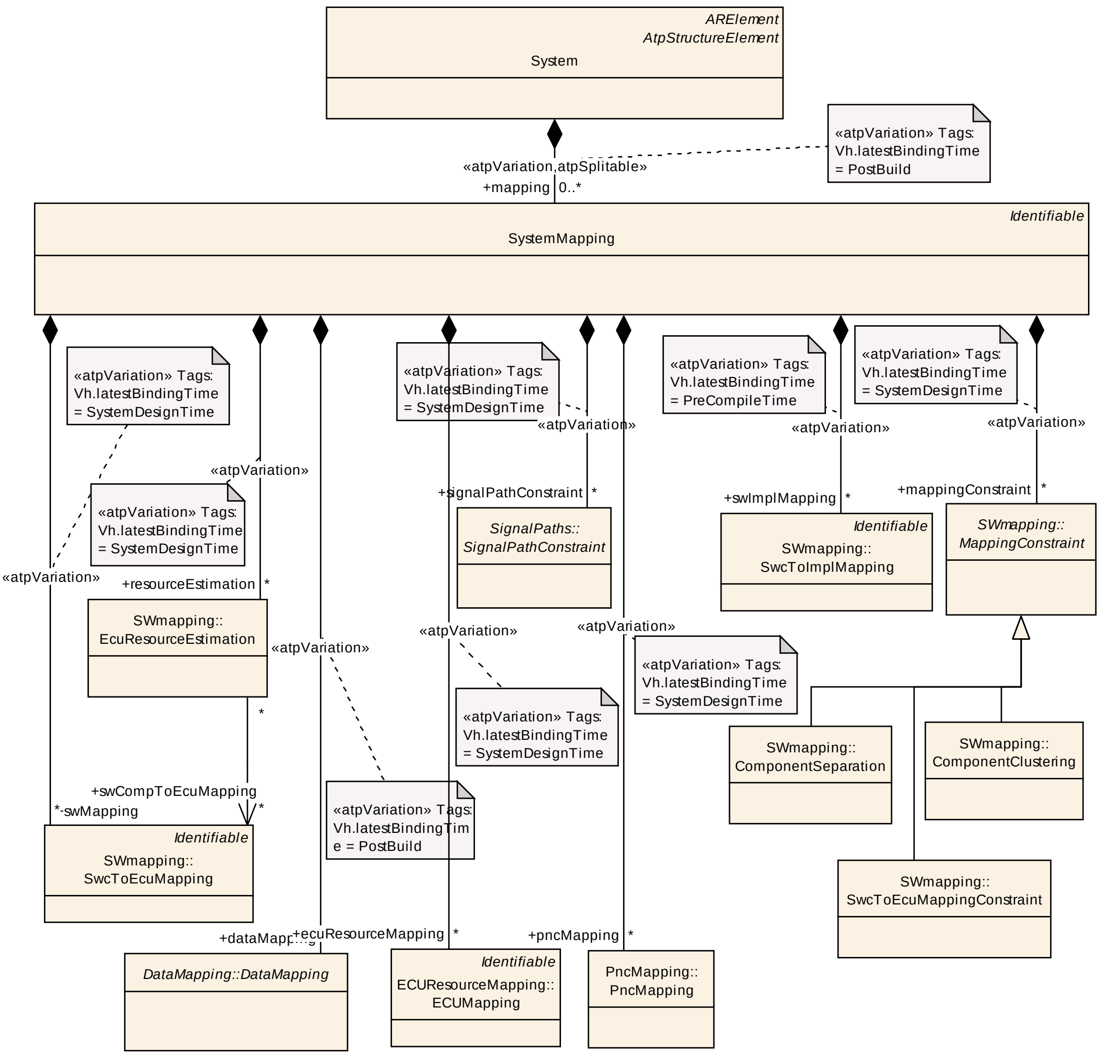

Figure 4.1: Mapping Overview (Mapping)

The following mappings are defined:

- The SwCompToEcuMapping meta-class maps one or several SwComponentPrototypes to ECUs. In the System Constraint Description it is possible to predefine the mapping of SwComponentPrototypes to ECUs. The predefinition limits the system architect's freedom to map software components to arbitrary ECUs. After the system generation in the System Configuration Description, all atomic software components that are directly or indirectly part of the top level composition must be mapped with this mapping rule. Software component mapping is described in detail in chapter 4.1.
- The SwCompToImplMapping meta-class is used to assign one Implementation to one or more SwComponentPrototypes (see chapter 4.1.2).
- The MappingConstraint meta-class is used to define constraints that constrain the mapping of software components. It's sub-classes allow to constraint which SwComponentPrototypes must be mapped together on the same ECU (ComponentClustering) and which must not be mapped to the same ECU (ComponentSeparation). The mapping constraints are described in detail in chapter 4.1.3.
- The DataMapping meta-class is used to map VariableDataPrototypes and ClientServerOperations in software component ports (i.e. the data exchanges between software components) to signals. The data mapping is described in detail in chapter 4.2.
- The SignalPathConstraint meta-class is used to define which specific way a signal (data element or client server operation arguments) between two Software Components should take in the network without defining in which frame and with which timing it is transmitted. This Signal Path Constraint is introduced in chapter 4.2.2.
- The ECUResourceMapping meta-class is used to map the hardware related topology elements onto their counterpart definitions in the ECU Resource Template (see chapter 2.4).
- Finally, meta-class EcuResourceEstimation specifies the resource estimation for RTE and basic software (see chapter 4.3).

Class	SystemMapping			
Package	M2::AUTOSARTemplates::SystemTemplate			
Note	The system mapping aggregates all mapping aspects (mapping of SW components to ECUs, mapping of data elements to signals, and mapping constraints).			
Base	ARObject,Identifiable,MultilanguageReferrable,Referrable			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
dataMapping	DataMapping	*	aggr	The data mappings defined. **Stereotypes:** atpVariation **Tags:** Vh.latestBindingTime=PostBuild
ecuResourceMapping	ECUMapping	*	aggr	Mapping of hardware related topology elements onto their counterpart definitions in the ECU Resource Template. atpVariation: The ECU Resource type might be variable. **Stereotypes:** atpVariation **Tags:** Vh.latestBindingTime=SystemDesignTime
mappingConstraint	MappingConstraint	*	aggr	Constraints that limit the mapping freedom for the mapping of SW components to ECUs. **Stereotypes:** atpVariation **Tags:** Vh.latestBindingTime=SystemDesignTime
pncMapping	PncMapping	*	aggr	**Stereotypes:** atpVariation**Tags:** Vh.latestBindingTime=SystemDesignTime
resourceEstimation	EcuResourceEstimation	*	aggr	Resource estimations for this set of mappings, zero or one per ECU instance. atpVariation: Used ECUs are variable. **Stereotypes:** atpVariation **Tags:** Vh.latestBindingTime=SystemDesignTime
signalPathConstraint	SignalPathConstraint	*	aggr	Constraints that limit the mapping freedom for the mapping of data elements to signals. **Stereotypes:** atpVariation **Tags:** Vh.latestBindingTime=SystemDesignTime
swImplMapping	SwcToImplMapping	*	aggr	The mappings of AtomicSoftwareComponentInstances to Implementations. atpVariation: Derived, because SwcToEcuMapping is variable. **Stereotypes:** atpVariation **Tags:** Vh.latestBindingTime=PreCompileTime
swMapping	SwcToEcuMapping	*	aggr	The mappings of SW components to ECUs. atpVariation: SWC shall be mapped to other ECUs. **Stereotypes:** atpVariation **Tags:** Vh.latestBindingTime=SystemDesignTime

Table 4.1: SystemMapping

— AUTOSAR CONFIDENTIAL —

4.1 Software Component Mapping

A fundamental concept of AUTOSAR is that SW components may be developed independently of a specific ECU hardware, and can be mapped to an ECU in the AUTOSAR System Generation Process. The System Constraint Description acts as an input to this System Generation Phase. Nevertheless, there may be some SW components which are already mapped due to previous iterations of the system generation step, and there may be system constraints that limit the system architect's freedom to map SW components to arbitrary ECUs. In the following, the individual elements are described in more detail.

4.1.1 SW Component to ECU Mapping

With `SwcToEcuMapping` element it is possible to express the mapping of `SwComponentPrototypes` to one `ECUInstance` or optional to individual `ProcessingUnits` residing in this ECU. An optional assignment to defined `EcuPartitions` (memory partitions) is also possible, as well as the assignment of Sensor/Actuator `SwComponentPrototypes` to Sensor/Actuator `HwElements`.

The mapping to cores and memory partitions enables to express the architectural requirements/constraints, especially related to safety. For example, it may be required that some SWCs shall run on different partitions or cores in the same ECU, or shall run on different ECUs. Figure 4.2 shows this structure. The predefinition will force the system generator to use the specified mapping.

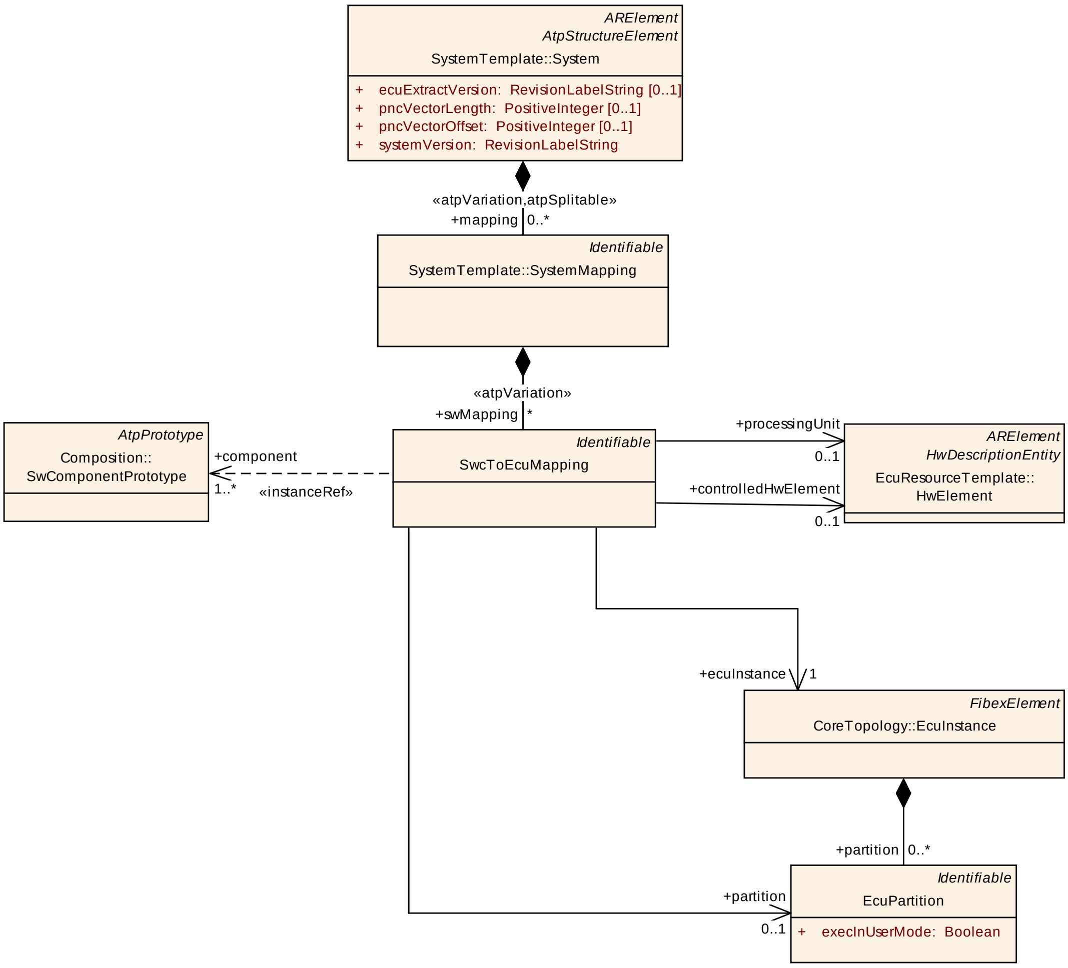

Figure 4.2: SW component to ECU mapping (SwcToEcuMapping)

For each `EcuInstance` which participates in the mapping, exactly one `SwCompToEcuMapping` shall be present in the `SystemMapping`: This `SwCompToEcuMapping` collects a list of all `SwComponentPrototypes` that shall be deployed onto the associated `EcuInstance`.

`SwCompToEcuMapping` may map either prototypes of `AtomicSoftwareComponentType` or those of `CompositionSwComponentType`. In case a prototype of an atomic Software Components is mapped, the mapping is unconditional. If on the other hand a mapped `SwComponentPrototype` refers to a `CompositionSwComponentType`, the mapping is applied to any inner `SwComponentPrototype` recursively; however, it may be overwritten by additional `SwCompToEcuMapping` mapping inner `SwComponentPrototype` to different `EcuInstances`.

Usually a particular component prototype can be mapped explicitly to at most one ECU in a given system (leaving aside variant handling and the implicit mapping of "inner" prototypes mentioned above) but there are two exceptions:

- A prototype of a `ParameterSwComponentType` can be mapped to more than one ECU. This is required, because this special component does not communicate over the network, so that a copy of the prototype has to be created on each ECU were it is required.

- Likewise, a prototype of an ServiceProxySwComponentType can be mapped to several ECUs even if it appears only once in the VFB system, because a prototype of this special component is required on each ECU, for which local Services are addressed via the proxy.

The restriction to zero or one SwCompToEcuMapping per EcuInstance holds for *System Configurations* where no variant handling is used. If Software Mapping is to be variable, there will be zero or one SwCompToEcuMapping per EcuInstance for the invariant mapping and additionally zero or one SwCompToEcuMappings for each Variation Condition.

[constr_3021] Mapping of SensorActuatorSwComponents to SensorActuator HwElements ⌈Only SwComponentPrototypes that are typed by SensorActuatorSwComponentType shall be mapped to a HwElement with category SensorActuator via the controlledHwElement relation. ⌋

The following table describes the SwcToEcuMapping in detail.

Class	**SwcToEcuMapping**			
Package	M2::AUTOSARTemplates::SystemTemplate::SWmapping			
Note	Map software components to a specific ECU Instance and optionally to a processing unit and to an EcuPartition. Per ECUInstance/ProcessingUnit/EcuPartition/SensorActuator only one SwcToEcuMapping shall be used.			
Base	ARObject,Identifiable,MultilanguageReferrable,Referrable			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**
component	SwComponentPrototype	1..*	iref	References to the software component instances that are mapped to the referenced ECUInstance. If the component prototype referenced is a composition, this indicates that all atomic software components within the composition are mapped to the ECU. If there is aditionally a mapping of some SwComponentPrototype INSIDE the Composition to another ECU Instance the inner mapping overrides the outer mapping.
controlledHwElement	HwElement	0..1	ref	Optional mapping of SwComponentPrototypes that are typed by SensorActuatorSwComponentType to a HwElement with category SensorActuator.
ecuInstance	EcuInstance	1	ref	EcuInstance is a reference to an ECU Instance description
partition	EcuPartition	0..1	ref	An optional mapping of SWCs to Partitions. With this mapping an OEM has the option to predefine an allocation in the System Design phase. The final and complete assignment is described in the OS Configuration.
processingUnit	HwElement	0..1	ref	Optional mapping of software components to individual microcontroller cores residing in one ECU. A microcontroller core is described in the ECU Resource Template by the HwElement of HwCategory ProcessingUnit.

Table 4.2: SwcToEcuMapping

4.1.2 Software Component to Implementation Mapping

As several implementations may exist for the same `AtomicSwComponentType`, it needs to be decided on and specified which instances of a given `AtomicSoftwareComponentType` are mapped to which `Implementation`. According to the AUTOSAR Methodology this information can either be added within the `Configure System` activity, or later when the RTE part is configured during `Configure ECU` phase. If the mapping is done in System Configuration, a `SwcToImplMapping` is being used for assigning one `Implementation` to one or more instances of `SwComponentPrototype` relating to the same `AtomicSwComponentType`. This is illustrated in Figure 4.3.

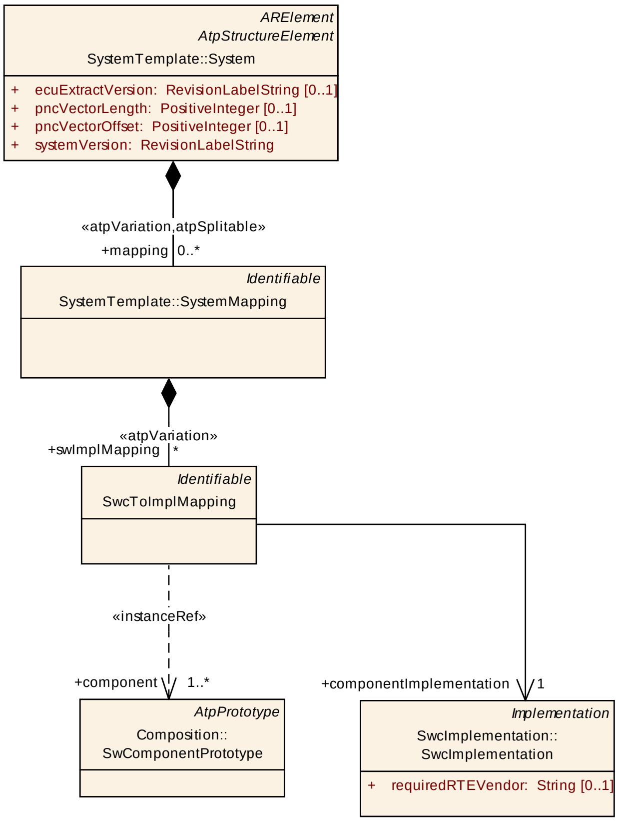

Figure 4.3: SW Component to Implementation mapping (SwcToImplMapping)

[constr_3002] valid swcToImplMapping ⌈The referenced `SwcImplementation` refers to a `SwcInternalBehavior` that is part of a `AtomicSwComponentType`. The same `AtomicSwComponentType` shall be the type of the referenced `SwComponentPrototype`.

SwcToImplMapping.componentImplementation.behavior.component == SwcToImplMapping.component.type ⌋

The following table contains the detailed description of SwcToImplMapping:

Class	**SwcToImplMapping**			
Package	M2::AUTOSARTemplates::SystemTemplate::SWmapping			
Note	Map instances of an AtomicSwComponentType to a specific Implementation.			
Base	ARObject,Identifiable,MultilanguageReferrable,Referrable			
Attribute	*Datatype*	*Mul.*	*Kind*	*Note*
component	SwComponentPrototype	1..*	iref	Reference to the software component instances that are being mapped to the specified Implementation. The targeted SwComponentPrototype needs be of the AtomicSwComponentType being implemented by the referenced Implementation.
componentImplementation	SwcImplementation	1	ref	Reference to a specific Implementation description. Implementation to be used by the specified SW component instance. This allows to achieve more precise estimates for the resource consumption that results from mapping the instance of an atomic SW component onto an ECU.

Table 4.3: SwcToImplMapping

4.1.3 Software Component Mapping Constraints

In contrast to the mapping description described in the previous chapters, mapping constraints allow to define invariants that have to be fulfilled by a valid mapping. They are aggregated in the `MappingConstraint` element as introduced in chapter 4 and depicted Figure 4.1. This chapter describes which mapping constraints can be described in the System Constraint Description. The description of this meta-class can be found in the following table:

Class	**MappingConstraint (abstract)**			
Package	M2::AUTOSARTemplates::SystemTemplate::SWmapping			
Note	Different constraints that may be used to limit the mapping of SW components to ECUs.			
Base	ARObject			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
introduction	Documentation Block	0..1	aggr	This represents introductory documentation about the mapping constraint.

Table 4.4: MappingConstraint

The two constraints (`ComponentClustering` and `ComponentSeparation`) shown in Figure 4.4 express the restrictions that Software Components impose each other when performing the mapping onto the ECUs. In fact, before the mapping process begins, it can be useful to impose the allocation of a predefined set of SW components onto the same ECU, especially if such a set is tightly linked from a functional point of view. In the same way, two critical SW components, performing some kind of redundancy, may be not suitable to run both on the same ECU. Thus, we call these two kinds of mapping constraints, respectively, `ComponentClustering` and `ComponentSeparation`.

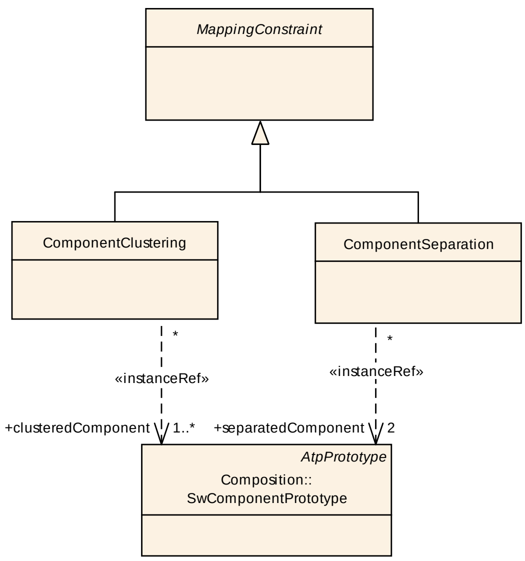

Figure 4.4: Details on ComponentClustering and ComponentSeparation (SwcClustering)

4.1.3.1 ComponentClustering

The `ComponentClustering` constraint (also, *clustering*) is to be used for expressing that a certain set of SW components (atomic or not) must be mapped (allocated) onto the same ECU. This is some kind of "execute together on same ECU" constraint.

The semantic of the clustering constraint is straightforward if all concerned SW components are atomic. Otherwise, it shall be interpreted as follows: all of the atomic SW components making up the composition must be mapped together onto the same ECU together with all other SW components (atomic or not) affected by the constraint. This also means that a *clustering* constraint can also refer to only a single composition.

A *clustering* constraint is part of a `MappingConstraint` element and it must refer to one or more `SwComponentPrototype` elements, representing the instances of the SW component(s) that must be mapped together.

Class	**ComponentClustering**			
Package	M2::AUTOSARTemplates::SystemTemplate::SWmapping			
Note	Constraint that forces the mapping of all referenced SW component instances to the same ECU			
Base	ARObject,MappingConstraint			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
clusteredComponent	SwComponentPrototype	1..*	iref	Reference to the components that have to be mapped together.

Table 4.5: ComponentClustering

4.1.3.2 ComponentSeparation

The `ComponentSeparation` constraint (also, *separation*) is to be used for expressing that two SW components (atomic or not) shall not be mapped (allocated) onto the same ECU. This is some kind of “do not execute together on same ECU” constraint.

The semantic of the separation constraint is straightforward if one or both SW components are atomic. Otherwise, it shall be interpreted as follows: any of the atomic SW components making up the first composition, must not be mapped onto the same ECU with any atomic SW component from the second composition. As a consequence, and to preserve consistency, an atomic SW component instance cannot be part of two compositions concerned by the same separation constraint, i.e. the two compositions have to be disjoint with regards to component instances[1].

[1]The only case where a component instance could be in both sets is if the `ComponentSeparation` refers to two elements where one of them is a substructure of the other. Consider the case that Atomic SW Component A is aggregated by composition B, which in turn is aggregated by composition C. Then instance A is both in B and C. It is not a good idea to formulate a separation constraint stating that B and C should not be on the same ECU.

A *separation* constraint is part of a `MappingConstraint` element and it must refer to two `SwComponentPrototype` elements, representing the two SW component instances that must not be allocated together.

Class	**ComponentSeparation**			
Package	M2::AUTOSARTemplates::SystemTemplate::SWmapping			
Note	Constraint that forces the two referenced SW components (called A and B in the following) not to be mapped to the same ECU. If a SW component (e.g. A) is a composition, none of the atomic SW components making up the A composition must be mapped together with any of the atomic SW components making up the B composition. Furthermore, A and B must be disjoint.			
Base	ARObject,MappingConstraint			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**
separatedComponent	SwComponentPrototype	2	iref	The two components that have to be mapped to different ECUs

Table 4.6: ComponentSeparation

[constr_3004] Clustering and separation must be exclusive ⌈Clustering and separation must be exclusive, i.e. it SHALL NOT be possible that two `ComponentPrototypes` A and B are associated by a `ComponentCluster` and by a `ComponentSeparation`.⌋

4.1.3.3 SwcToEcuMappingConstraint

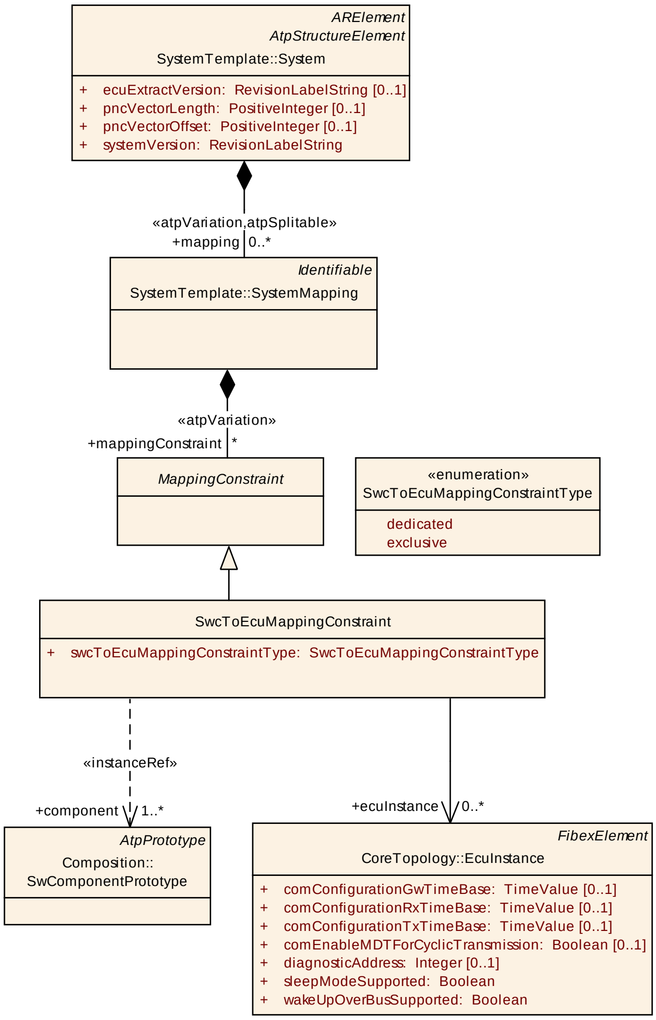

Figure 4.5: Dedicated and exclusive Mapping of SWC to ECUs

The `SwcToEcuMappingConstraint` shown in Figure 4.5 allows to restrict the mapping of SW components to ECUs. If the `swcToEcuMappingConstraintType` is set to `dedicated`, the constraint expresses that the mapping of specific SW components is only allowed to one of a number of dedicated ECUs. The mapping to other ECUs is not allowed. When the system generator performs the mapping of software components to ECUs it has to take these constraints into account.

If the `swcToEcuMappingConstraintType` is set to `exclusive`, it means that the referenced software components cannot be mapped to the referenced ECUs.

With these kinds of constraints, no fixed mapping of a software component to an ECU is performed. Instead, they can be seen as invariants that have to be fulfilled when the actual SWC mapping using SwcToEcuMapping is performed.

Class	**SwcToEcuMappingConstraint**			
Package	M2::AUTOSARTemplates::SystemTemplate::SWmapping			
Note	The System Constraint Description has to describe dedicated and exclusive mapping of SW-Cs to one or more ECUs. Dedicated mapping means that the SW-C can only be mapped to the ECUs it is dedicated to. Exclusive Mapping means that the SW-C cannot be mapped to the ECUs it is excluded from.			
Base	ARObject,MappingConstraint			
Attribute	*Datatype*	*Mul.*	*Kind*	*Note*
component	SwComponentPrototype	1	iref	Reference to SwComponentPrototypes for which the dedicated or exclusive mapping is defined.
ecuInstance	EcuInstance	*	ref	If the dedicated mapping is described, the SwComponentPrototypes can only be mapped to these referenced ECUInstances. If the exclusive mapping is described, the SwComponentPrototypes cannot be mapped to these referenced ECUInstances.
swcToEcuMappingConstraintType	SwcToEcuMappingConstraintType	1	attr	This attribute determines if dedicated or exclusive mapping is used.

Table 4.7: SwcToEcuMappingConstraint

Enumeration	**SwcToEcuMappingConstraintType**
Package	M2::AUTOSARTemplates::SystemTemplate::SWmapping
Note	There are two different SwcToEcuMapping constraints: dedicated mapping and exclusive mapping.
Literal	*Description*
dedicated	Dedicated mapping means that the SW-C can only be mapped to the ECUs it is dedicated to.
exclusive	Exclusive mapping means that the SW-C cannot be mapped to the ECUs it is excluded from.

Table 4.8: SwcToEcuMappingConstraintType

4.2 Data Mapping

The data mapping description may either be mapping of client server communication or sender receiver communication (see Figure 4.6). It is used to map `VariableDataPrototypes` or `ClientServerOperations` of SW Component Ports to `SystemSignals`.

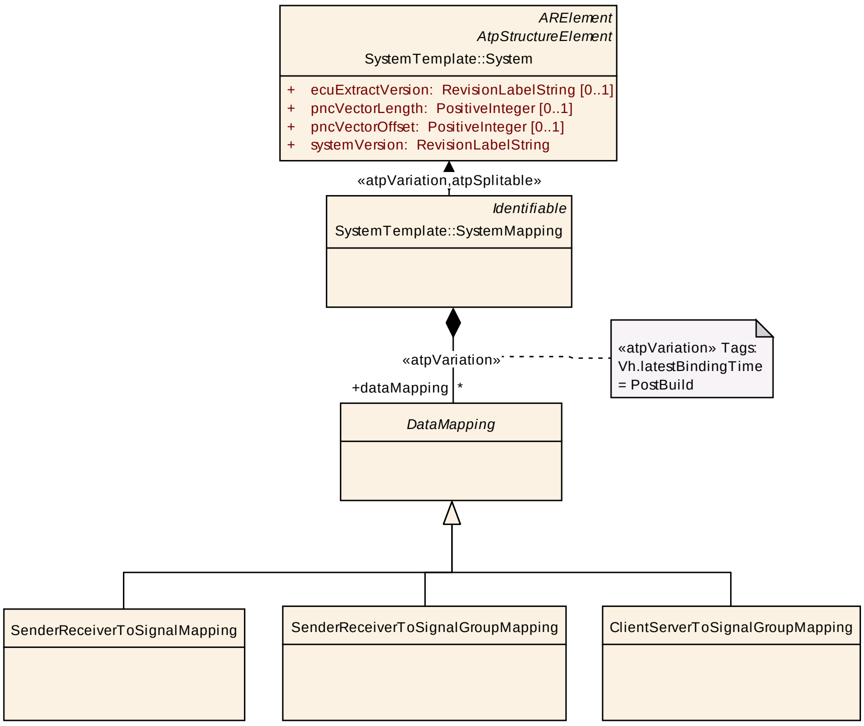

Figure 4.6: Overview: Data Mapping Description (DataMappingOverview)

`SystemSignals` represent `VariableDataPrototypes` and `ClientServerOperations` in the communication description. The `SystemSignals` are unique per System and can be defined independently of frames and communication clusters. This chapter describes how the `VariableDataPrototypes` and `ClientServerOperations` are mapped onto `SystemSignals`. The Communication chapter (5) describes how the `SystemSignals` are mapped into Pdus and Frames, implementing the actual inter-ECU communication.

Class	SystemSignal			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::FibexCore::CoreCommunication			
Note	The system signal represents the communication system's view of data exchanged between SW components which reside on different ECUs. The system signals allow to represent this communication in a flattened structure, with exactly one system signal defined for each data element prototype sent and received by connected SW component instances. In case the System Description doesn't use a complete Software Component Description (VFB View) the data mapping of Variable Data Prototypes or Client Server Operations on SystemSignals needs not to be defined. This supports the inclusion of legacy signals. **Tags:** atp.recommendedPackage=SystemSignals			
Base	ARElement,ARObject,CollectableElement,Identifiable,Multilanguage Referrable,PackageableElement,Referrable			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
dynamicLength	Boolean	1	attr	The length of dynamic length signals is variable in run-time. Only a maximum length of such a signal is specified in the configuration (attribute length in ISignal element).

Table 4.9: SystemSignal

In case that a VariableDataPrototype is transferred over the network a SystemSignal is being defined representing the VariableDataPrototype on the network. SystemSignals are unique in the sense that the same SystemSignal represents the same VariableDataPrototype system wide.

In case of 1:n communication the VariableDataPrototype in the ProvidePort of the SwComponentPrototype is still mapped to only one SystemSignal. In case of n:1 communications each sender needs to be represented by an different SystemSignal.

The different data mappings are described in the following chapters in detail.

4.2.1 Mapping of Variable Data Prototypes on System Signals

This chapter describes how VariableDataPrototype, being the units of information to be transported between providing and requiring ports, are mapped onto SystemSignals.

In the Software part of the System Template (3) a top-level RootSwCompositionPrototype is expressed by using AssemblySwConnectors and DelegationSwConnector to connect the PPortPrototypes and RPortPrototypes of SwComponentPrototypes with each other on the VFB-level.

Ultimately, each chain of SwConnectors leads to exactly one PPortPrototype. This PPortPrototype references a PortInterface, which may either be a SenderReceiverInterface or a ClientServerInterface. It is the task of system configuration to map each VariableDataPrototype or ArgumentPrototype contained in these Ports referenced by the SwConnector onto a SystemSignal. However, the same SystemSignal may satisfy more than one connector (1:n communication), and one connector may be implemented by several SystemSignals (e.g. one per VariableDataPrototype in the PortInterface being connected), so there is no 1:1 mapping between AssemblyConnectors and SystemSignals. Therefore, if one needs to find all SystemSignals implementing a particular AssemblyConnector, this requires a model query which compares the ProvidedPort end of the connector chain with the PortPrototype providing the VariableDataPrototype.

In the following sections, each reference to a VariableDataPrototype or ArgumentDataPrototype is of type Instance Reference [1]. This means it not only references the actual VariableDataPrototype, but additionally contains contextual references to the PortPrototype and the hierarchy of SwComponentPrototypes forming the individual instance context of the VariableDataPrototype. Therefore the above mentioned query requires a comparison of the full instance reference paths of the connector end and the PortPrototype context of the VariableDataPrototype to be mapped to the signal.

The following rules are valid for the mapping of Variable Data Prototypes and Client Server Operations on SystemSignals:

1) For each SystemSignal in a complete System Description exactly one data mapping shall be defined (P-Port or R-Port). Preference: P-Port

In a complete System Description, it is sufficient to refer to the VariableDataPrototype in the ProvidePort or the RequirePort to define the mapping of the communication between a provider and its receivers. This is possible since the connectors implicitly define which RequirePorts are connected to which ProvidePort. In case the System Description doesn't use a complete Software Component Description (VFB View) the data mapping needs not to be defined. This supports the inclusion of legacy signals.

2) In the System Extract/ECU Extract the missing data mappings on the complementary Sender/Receiver side needs to be supplemented.

In a System extract and ECU extract of the system description, where only the relevant parts of the SW compositions are defined, it is necessary to utilize the information from the complementary Port, if the corresponding Port is located on another ECU and thus is not part of the extract. This is described in more detail in chapter 8.2 and chapter 9.2.3. Therefore a data mapping can be provided on ProvidePorts and on RequirePorts.

3) Data mappings can be performed on compositions and on atomic SWCs.

4) During the creation of the ECU Extract (flattening) the existing data mappings on compositions needs to be transfered to the atomic SWCs.

In the OEM/Supplier Collaboration Scenario the outer shell of a Software Composition (an empty composition) is defined by an OEM and is delivered to a supplier. The Supplier adds the substructure to the Composition by adding atomic `SwComponentPrototypes` and `SwConnectors`. But the supplier must respect the predefined data mapping on the Software Composition. The OEM/Supplier Collaboration Scenario is described in chapter 8.1.

5) If a SW Composition is refined by the Supplier the mapped data elements of the composition shall not be mapped a second time in the internal substructure. In a subsequent ECU extract step according to rule 4) the mappings will be transferred to the inner components.

4.2.1.1 Mapping of Variable Data Prototypes with primitive datatypes on System Signals (Sender-Receiver Communication)

The `VariableDataPrototype` meta-class is defined in the SW Component Template. The datatype of the `VariableDataPrototype` may be a primitive one or a composite one. Primitive data types cannot be decomposed in other data types. The composite data types "array" and "record" provide the means to build new data types.

This chapter describes the relation between the `VariableDataPrototype` with primitive datatypes and the `SystemSignal` (see Figure 4.7). The primitive type mapping can also be used for the data mapping of UINT8-Arrays. This supports an optimized definition of the data mapping. More details can be found in section 4.2.1.2 and section 4.2.1.3.

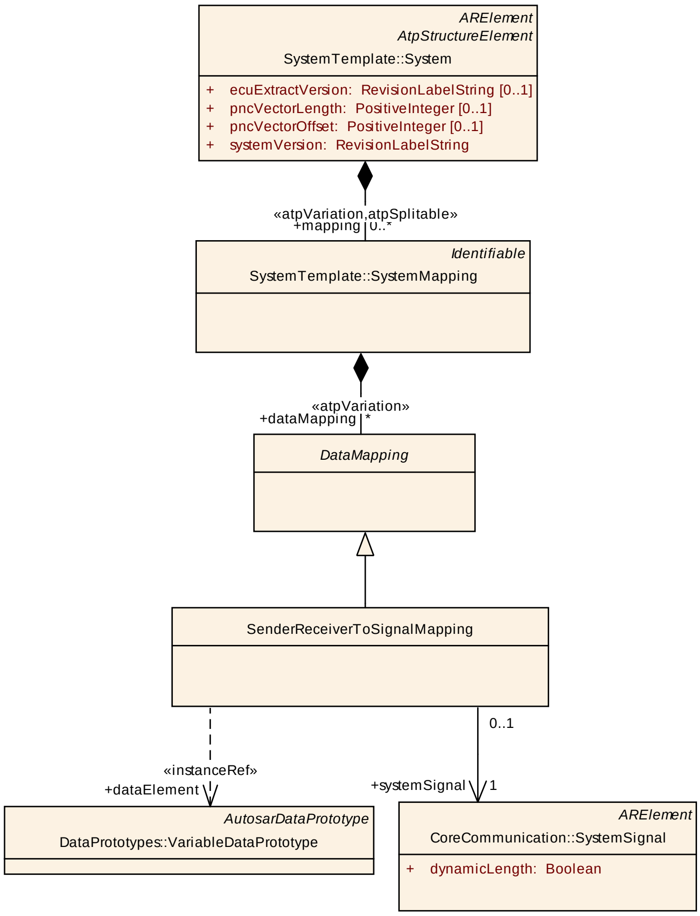

Figure 4.7: Mapping of data elements with primitive datatypes (SenderRecPrimitiveTypeMapping)

Class	**SenderReceiverToSignalMapping**			
Package	M2::AUTOSARTemplates::SystemTemplate::DataMapping			
Note	Mapping of a sender receiver communication data element with a primitive datatype to a signal. If the data element has to be transmitted to several receivers there is still exactly one mapping defined. In case of 1:n communication the VariableDataPrototype in the ProvidePort of the SwComponentPrototype is still mapped to only one SystemSignal.			
Base	ARObject,DataMapping			
Attribute	*Datatype*	*Mul.*	*Kind*	*Note*
dataElement	VariableDataPrototype	1	iref	Reference to the data element, which ought to be sent over the Communication bus.
systemSignal	SystemSignal	1	ref	Reference to the system signal used to carry the data element.

Table 4.10: SenderReceiverToSignalMapping

4.2.1.2 Mapping of Variable Data Prototypes with composite datatypes on Signal Groups (Sender-Receiver Communication)

This chapter describes the mapping of `VariableDataPrototype` with composite datatypes to `SystemSignals`.

The RTE is required to treat AUTOSAR signals transmitted using sender-receiver communication atomically. To achieve this, the "signal group" mechanisms shall be utilized. It is not possible to map a `VariableDataPrototype` with a composite datatype directly to a `SystemSignal`. The complex data type must be decomposed into single signals. As this set of single signals has to be treated as atomic, it is placed in a "signal group". There is one exception to this rule: it is allowed to map an array `VariableDataPrototype` consisting of UINT8 elements to exactly one `SystemSignal` via the `SenderReceiverToSignalMapping`. A UINT8 element may be a String or an array that contains array elements of Integer type with range 0..255.

In the ECU Configuration of the AUTOSAR COM module such a `SystemSignal` will be mapped to a COM Signal with the `ComSignalType` UINT8_N.

If the "signal group" mechanisms is used each "primitive" `RecordElement` or `ArrayElement` in the context of the complex element is mapped to a `SystemSignal`. The `VariableDataPrototype` that is referenced by `SenderReceiverToSignalGroupMapping` can be typed by an `ApplicationDataType` or by an `ImplementationDataType`. This type decides which reference is used within the `SenderRecRecordElementMapping` and `SenderRecArrayElementMapping`.

Complex `VariableDataPrototypes` may nest within other complex `VariableDataPrototypes`. Each `PrimitiveDataType` of such nested complex `VariableDataPrototypes` will be one `SystemSignal` in the System Description.

The relationship between the `SystemSignal` and the `VariableDataPrototype` is provided in the `SenderReceiverToSignalGroupMapping` (see Figure 4.8).

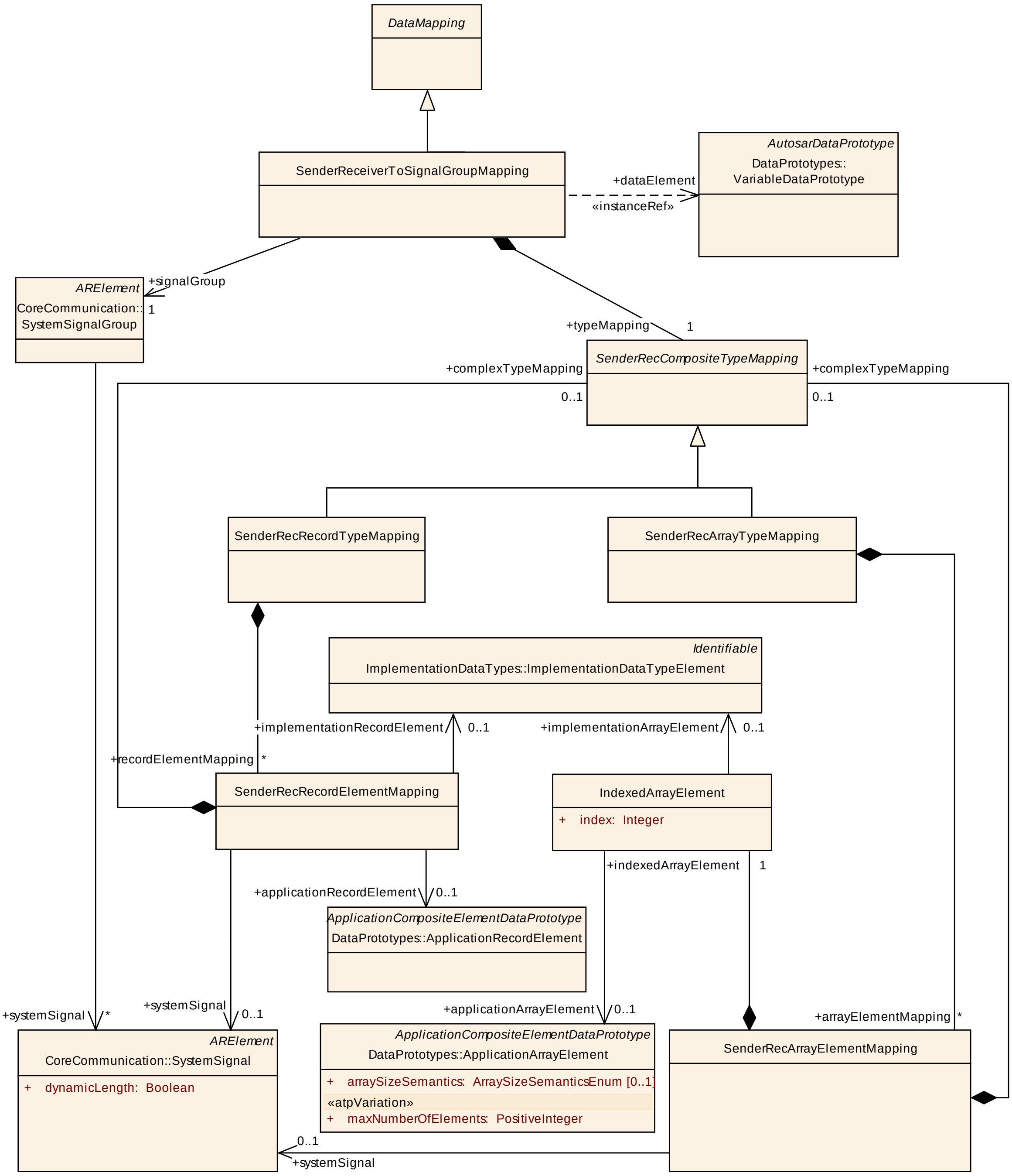

Figure 4.8: Mapping of data elements with composite datatypes (SenderRecCompositeTypeMapping)

[constr_3000] valid `SenderRecCompositeTypeMappings` ⌈SenderReceiverToSignalGroupMapping.signalGroup.systemSignal shall point to each `SystemSignal` being mapped within the context of `SenderReceiverToSignalGroupMapping`. ⌋

Class	SenderReceiverToSignalGroupMapping			
Package	M2::AUTOSARTemplates::SystemTemplate::DataMapping			
Note	Mapping of a sender receiver communication data element with a composite datatype to a signal group.			
Base	ARObject,DataMapping			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**
dataElement	VariableDataPrototype	1	iref	Reference to a data element with a composite datatype which is mapped to a signal group.
signalGroup	SystemSignalGroup	1	ref	Reference to the signal group, which contain all primitive datatypes of the composite type
typeMapping	SenderRecCompositeTypeMapping	1	aggr	The CompositeTypeMapping maps the the ApplicationArrayElements and ApplicationRecordElements to Signals of the SignalGroup.

Table 4.11: SenderReceiverToSignalGroupMapping

Class	SenderRecCompositeTypeMapping (abstract)			
Package	M2::AUTOSARTemplates::SystemTemplate::DataMapping			
Note	Two mappings exist for the composite data types: "ArrayTypeMapping" and "RecordTypeMapping". In both, a primitive datatype will be mapped to a system signal. But it is also possible to combine the arrays and the records, so that an "array" could be an element of a "record" and in the same manner a "record" could be an element of an "array". Nesting these data types is also possible. If an element of a composite data type is again a composite one, the "CompositeTypeMapping" element will be used one more time (aggregation between the ArrayElementMapping and CompositeTypeMapping or aggregation between the RecordElementMapping and CompositeTypeMapping).			
Base	ARObject			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**
–	–	–	–	–

Table 4.12: SenderRecCompositeTypeMapping

Class	SenderRecArrayTypeMapping			
Package	M2::AUTOSARTemplates::SystemTemplate::DataMapping			
Note	If the ApplicationCompositeDataType is an Array, the "ArrayTypeMapping" will be used.			
Base	ARObject,SenderRecCompositeTypeMapping			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**
arrayElementMapping	SenderRecArrayElementMapping	*	aggr	Each ApplicationArrayElement must be mapped on a SystemSignal.

Table 4.13: SenderRecArrayTypeMapping

— AUTOSAR CONFIDENTIAL —

Class	SenderRecRecordTypeMapping			
Package	M2::AUTOSARTemplates::SystemTemplate::DataMapping			
Note	If the ApplicationCompositeDataType is a Record, the "RecordTypeMapping" will be used.			
Base	ARObject,SenderRecCompositeTypeMapping			
Attribute	Datatype	Mul.	Kind	Note
recordElementMapping	SenderRecRecordElementMapping	*	aggr	Each ApplicationRecordElement must be mapped on a SystemSignal.

Table 4.14: SenderRecRecordTypeMapping

Class	SenderRecRecordElementMapping			
Package	M2::AUTOSARTemplates::SystemTemplate::DataMapping			
Note	Mapping of a primitive record element to a SystemSignal. If the VariableDataPrototype that is referenced by SenderReceiverToSignalGroupMapping is typed by an ApplicationDataType the reference applicationRecordElement shall be used. If the VariableDataPrototype is typed by the ImplementationDataType the reference implementationRecordElement shall be used. Either the implementationRecordElement or applicationRecordElement reference shall be used. If the element is composite, there will be no mapping to the SystemSignal (multiplicity 0). In this case the RecordElementMapping element will aggregate the complexTypeMapping element. In that way also the composite datatypes can be mapped to SystemSignals.			
Base	ARObject			
Attribute	Datatype	Mul.	Kind	Note
applicationRecordElement	ApplicationRecordElement	0..1	ref	Reference to an ApplicationRecordElement in the context of the dataElement or in the context of a composite element. This reference shall only be used if the VariableDataPrototype that is referenced by the SenderReceiverToSignalGroupMapping.dataElement is typed by an ApplicationDataType.
complexTypeMapping	SenderRecCompositeTypeMapping	0..1	aggr	This aggregation will be used if the element is composite.
implementationRecordElement	ImplementationDataTypeElement	0..1	ref	Reference to an ImplementationRecordElement in the context of the dataElement or in the context of a composite element. This reference shall only be used if VariableDataPrototype that is referenced by the SenderReceiverToSignalGroupMapping.dataElement is typed by an ImplementationDataType.
systemSignal	SystemSignal	0..1	ref	Reference to the system signal used to carry the primitive ApplicationRecordElement.

Table 4.15: SenderRecRecordElementMapping

Class	SenderRecArrayElementMapping			
Package	M2::AUTOSARTemplates::SystemTemplate::DataMapping			
Note	The SenderRecArrayElement may be a primitive one or a composite one. If the element is primitive, it will be mapped to the SystemSignal (multiplicity 1). If the VariableDataPrototype that is referenced by SenderReceiverToSignalGroupMapping is typed by an ApplicationDataType the reference to the ApplicationArrayElement shall be used. If the VariableDataPrototype is typed by the ImplementationDataType the reference to the ImplementationArrayElement shall be used. If the element is composite, there will be no mapping to the SystemSignal (multiplicity 0). In this case the ArrayElementMapping element will aggregate the TypeMapping element. In that way also the composite datatypes can be mapped to SystemSignals. Regardless whether composite or primitive array element is mapped the indexed element always needs to be specified.			
Base	ARObject			
Attribute	Datatype	Mul.	Kind	Note
complexTypeMapping	SenderRecCompositeTypeMapping	0..1	aggr	This aggregation will be used if the element is composite.
indexedArrayElement	IndexedArrayElement	1	aggr	Reference to an indexed array element in the context of the dataElement or in the context of a composite element.
systemSignal	SystemSignal	0..1	ref	Reference to the system signal used to carry the primitive ApplicationArrayElement.

Table 4.16: SenderRecArrayElementMapping

Class	IndexedArrayElement			
Package	M2::AUTOSARTemplates::SystemTemplate::DataMapping			
Note	This element represents exactly one indexed element in the array. Either the applicationArrayElement or implementationArrayElement reference shall be used.			
Base	ARObject			
Attribute	Datatype	Mul.	Kind	Note
applicationArrayElement	ApplicationArrayElement	0..1	ref	Reference to an ApplicationArrayElement in an array. This reference shall only be used if the referenced context element (VariableDataPrototype that is referenced by the SenderReceiverToSignalGroupMapping or ArgumentDataPrototype that is referenced by the ClientServerCompositeTypeMapping) is typed by an ApplicationDataType.
implementationArrayElement	ImplementationDataTypeElement	0..1	ref	Reference to an ImplementationDataTypeElement in an array. This reference shall only be used if the referenced context element (VariableDataPrototype that is referenced by the SenderReceiverToSignalGroupMapping or ArgumentDataPrototype that is referenced by the ClientServerCompositeTypeMapping) is typed by an ImplementationDataType.
index	Integer	1	attr	Position of an element in an array.

Table 4.17: IndexedArrayElement

Figure 4.9 shows a mapping example for nested complex datatypes.

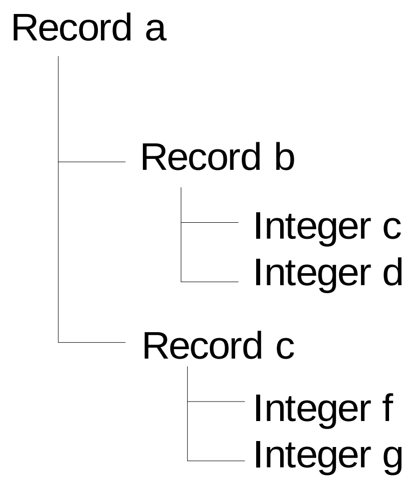

Figure 4.9: Mapping example for nested complex datatypes

The first `SenderRecRecordElementMapping` for RECORD a does not contain a reference to a `SystemSignal` because signals apply only to atomic data items. Instead it contains a `SenderRecCompositeTypeMapping` which in turn contains a `SenderRecRecordElementMapping` for INTEGER c (which does have a corresponding signal).

4.2.1.3 Mapping of Client Server Operations to Signal Groups

The Client/Server interfaces aggregate a number of Client Server operations. Each description of an operation consists of the description of its arguments. Furthermore, the RTE is responsible to map a response to the corresponding request. For this mapping transaction handles are used. The transaction handle contain a client identifier and a sequence counter.

The arguments, application errors, client identifier and sequence counter of an operation are mapped to `SystemSignals` of two dedicated `SystemSignalGroup` elements; one for the request and one for the response. The RTE Client Server Protocol is used to provide a specific semantics to each of these `SystemSignalGroups` and `SystemSignals`, also those which are introduced only to support the protocol. This is described in more detail in [15].

The relationship between the `SystemSignals` and the `Arguments` is provided in the `ClientServerToSignalGroupMapping` (see Figure 4.10).

The datatype of an argument may be a primitive one or a composite one. Each primitive argument will be mapped directly onto one `SystemSignal` by `ClientServerPrimitiveTypeMapping`. The complex data type shall be decomposed into single signals by `ClientServerCompositeTypeMapping`. There is one exception to this rule: it is allowed to map an array `ArgumentDataPrototype` consisting of UINT8 elements to exactly one SystemSignal via the ClientServerPrimitiveTypeMapping. A UINT8 element may be a String or an array that contains array elements of Integer type with range 0..255.

The ArgumentDataPrototype that is referenced by ClientServerCompositeTypeMapping can be typed by an ApplicationDataType or by an ImplementationDataType. This type decides which reference is used within the ClientServerRecordElementMapping and ClientServerArrayElementMapping.

In a complete System Description, it is sufficient to refer to the operation in the ProvidePort to define the mapping of the communication between a provider and its receivers. This is possible since the connectors implicitly define which RequirePorts are connected to the ProvidePort. In an ECU extract/System Extract of the system description, where only the relevant parts of the SW compositions are defined, it is in some cases also necessary to refer to RequirePorts, if the corresponding ProvidePort is not part of the extract. This is described in more detail in chapter 8.2 for the System Extract and chapter 9.2.3 for the ECU Extract.

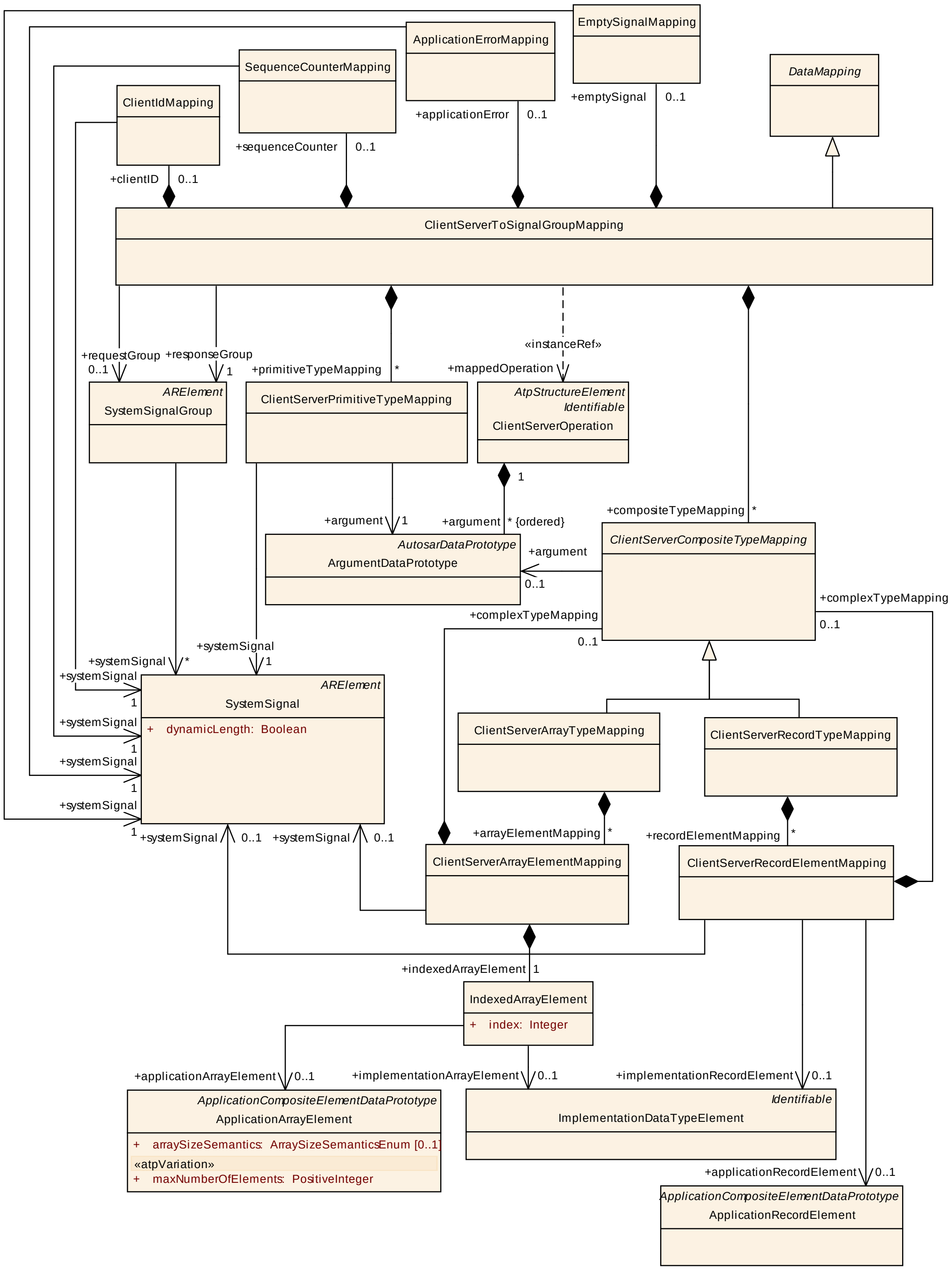

Figure 4.10: Operation Mapping (ClientServerOperationMapping)

[constr_3001] valid `ClientServerToSignalGroupMappings` ⌈ `SystemSignals` that are referenced by a `ClientServerArrayTypeMapping` or `ClientServerRecordTypeMapping` within the context of `ClientServerToSignalGroupMapping` shall also be referenced by ClientServerToSignal-

GroupMapping.requestGroup.systemSignal or ClientServerToSignalGroupMapping.responseGroup.systemSignal. ⌋

Class	**ClientServerToSignalGroupMapping**			
Package	M2::AUTOSARTemplates::SystemTemplate::DataMapping			
Note	Mapping of client server operation arguments to signals of a signal group. Arguments with a primitive datatype will be mapped via the "ClientServerPrimitiveTypeMapping" element. Arguments with composite datatypes will be mapped via the "CompositeTypeMapping" element.			
Base	ARObject,DataMapping			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**
applicationError	ApplicationErrorMapping	0..1	aggr	In client server communication, the server may return any value within the application error range.
clientID	ClientIdMapping	0..1	aggr	In case of a server on one ECU with multiple clients on other ECUs, the client server communication shall use different unique COM signals and signal groups for each client to allow the identification of the client associated with each system signal.
compositeTypeMapping	ClientServerCompositeTypeMapping	*	aggr	Mapping of arguments with composite datatypes.
emptySignal	EmptySignalMapping	0..1	aggr	An emptySignal is created if no actual data is configured for a client-server communication, but if the RTE shall send a SignalGroup to initiate the communication. An EmptySignalMapping shall only reference a SystemSignal that is referenced by an ISignal with length equal to zero.
mappedOperation	ClientServerOperation	1	iref	Reference to a Operation, which is mapped to a signal group.
primitiveTypeMapping	ClientServerPrimitiveTypeMapping	*	aggr	Mapping of an argument with a primitive datatype to a signal.
requestGroup	SystemSignalGroup	0..1	ref	Reference to the signal group which contains the references to request signals used to transport the IN and INOUT arguments of the operation.
responseGroup	SystemSignalGroup	1	ref	Reference to the signal group which contains the references to response signals used to transport the OUT and INOUT arguments of the operation.
sequenceCounter	SequenceCounterMapping	0..1	aggr	The purpose of sequence counters is to map a response to the correct request of a known client.

Table 4.18: ClientServerToSignalGroupMapping

[constr_3026] valid `EmptySignalMappings` ⌈ An `EmptySignalMapping` shall only reference a `SystemSignal` that is referenced by an `ISignal` with length equal to zero. ⌋

Class	**ClientIdMapping**			
Package	M2::AUTOSARTemplates::SystemTemplate::DataMapping			
Note	In case of a server on one ECU with multiple clients on other ECUs, the client server communication shall use different unique COM signals and signal groups for each client to allow the identification of the client associated with each system signal. The ClientId is mapped to the requestGroup and to the responseGroup.			
Base	ARObject			
Attribute	*Datatype*	*Mul.*	*Kind*	*Note*
systemSignal	SystemSignal	1	ref	Reference to the SystemSignal with the ClientID.

Table 4.19: ClientIdMapping

Class	**SequenceCounterMapping**			
Package	M2::AUTOSARTemplates::SystemTemplate::DataMapping			
Note	The purpose of sequence counters is to map a response to the correct request of a known client. The SequenceCounter is mapped to the requestGroup and to the responseGroup.			
Base	ARObject			
Attribute	*Datatype*	*Mul.*	*Kind*	*Note*
systemSignal	SystemSignal	1	ref	Reference to the SystemSignal with the SequenceCounter.

Table 4.20: SequenceCounterMapping

Class	**ApplicationErrorMapping**			
Package	M2::AUTOSARTemplates::SystemTemplate::DataMapping			
Note	In client server communication, the server may return any value within the application error range. The ApplicationError is mapped to the responseGroup.			
Base	ARObject			
Attribute	*Datatype*	*Mul.*	*Kind*	*Note*
systemSignal	SystemSignal	1	ref	Reference to the SystemSignal with the ApplicationError.

Table 4.21: ApplicationErrorMapping

Class	**EmptySignalMapping**			
Package	M2::AUTOSARTemplates::SystemTemplate::DataMapping			
Note	If no actual data is configured for a client server communication the element EmptySignalMapping shall be used. An EmptySignalMapping shall only reference a SystemSignal that is referenced by an ISignal with length equal to zero. In this case there shall be an "update-bit" configured. The EmptySignal can be mapped to the response group or to request group.			
Base	ARObject			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**
systemSignal	SystemSignal	1	ref	Reference to a SystemSignal with "signalLength" = 0 and an UpdateBit.

Table 4.22: EmptySignalMapping

Class	**ClientServerPrimitiveTypeMapping**			
Package	M2::AUTOSARTemplates::SystemTemplate::DataMapping			
Note	Mapping of an argument with a primitive datatype to a signal.			
Base	ARObject			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**
argument	ArgumentDataPrototype	1	ref	Reference to an argument in the context of the mappedOperation.
systemSignal	SystemSignal	1	ref	Reference to the system signal used to carry the argument

Table 4.23: ClientServerPrimitiveTypeMapping

Class	**ClientServerCompositeTypeMapping (abstract)**			
Package	M2::AUTOSARTemplates::SystemTemplate::DataMapping			
Note	Two mappings exist for the composite data types: "ArrayTypeMapping" and "RecordTypeMapping". In both, a primitive datatype will be mapped to a system signal. But it is also possible to combine the arrays and the records, so that an "array" could be an element of a "record" and in the same manner a "record" could be an element of an "array". Nesting these data types is also possible. If an element of a composite data type is again a composite one, the "CompositeTypeMapping" element will be used one more time (aggregation between the ArrayElementMapping and CompositeTypeMapping or aggregation between the RecordElementMapping and CompositeTypeMapping).			
Base	ARObject			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**
argument	ArgumentDataPrototype	0..1	ref	Reference to an argument in the context of the mappedOperation. Only ClientServerCompositeTypeMapping elements that are directly aggregated by the ClientServerToSignalGroupMapping shall contain this reference.

Table 4.24: ClientServerCompositeTypeMapping

Class	**ClientServerArrayTypeMapping**			
Package	M2::AUTOSARTemplates::SystemTemplate::DataMapping			
Note	If the ApplicationCompositeDataType is an Array, the "ArrayTypeMapping" will be used.			
Base	ARObject,ClientServerCompositeTypeMapping			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**
arrayElementMapping	ClientServerArrayElementMapping	*	aggr	Each ApplicationArrayElement must be mapped on a SystemSignal.

Table 4.25: ClientServerArrayTypeMapping

Class	**ClientServerArrayElementMapping**			
Package	M2::AUTOSARTemplates::SystemTemplate::DataMapping			
Note	The ApplicationArrayElement may be a primitive one or a composite one. If the element is primitive, it will be mapped to the "SystemSignal" (multiplicity 1). If the ArgumentDataPrototype that is referenced by ClientServerCompositeTypeMapping is typed by an ApplicationDataType the reference to the ApplicationArrayElement shall be used. If the VariableDataPrototype is typed by the ImplementationDataType the reference to the ImplementationArrayElement shall be used. If the element is composite, there will be no mapping to the "SystemSignal" (multiplicity 0). In this case the "ArrayElementMapping" Element will aggregate the "TypeMapping" Element. In that way also the composite datatypes can be mapped to SystemSignals. Regardless whether composite or primitive array element is mapped the indexed array element always needs to be specified.			
Base	ARObject			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**
complexTypeMapping	ClientServerCompositeTypeMapping	0..1	aggr	This aggregation will be used if the element is composite.
indexedArrayElement	IndexedArrayElement	1	aggr	Reference to an indexed array element in the context of the mappedOperation or in the context of a composite element.
systemSignal	SystemSignal	0..1	ref	Reference to the system signal used to carry the primitive ApplicationArrayElement.

Table 4.26: ClientServerArrayElementMapping

Class	**ClientServerRecordTypeMapping**			
Package	M2::AUTOSARTemplates::SystemTemplate::DataMapping			
Note	If the ApplicationCompositeDataType is a Record, the "RecordTypeMapping" will be used.			
Base	ARObject,ClientServerCompositeTypeMapping			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
recordElementMapping	ClientServerRecordElementMapping	*	aggr	Each ApplicationRecordElement must be mapped on a SystemSignal.

Table 4.27: ClientServerRecordTypeMapping

Class	**IndexedArrayElement**			
Package	M2::AUTOSARTemplates::SystemTemplate::DataMapping			
Note	This element represents exactly one indexed element in the array. Either the applicationArrayElement or implementationArrayElement reference shall be used.			
Base	ARObject			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
applicationArrayElement	ApplicationArrayElement	0..1	ref	Reference to an ApplicationArrayElement in an array. This reference shall only be used if the referenced context element (VariableDataPrototype that is referenced by the SenderReceiverToSignalGroupMapping or ArgumentDataPrototype that is referenced by the ClientServerCompositeTypeMapping) is typed by an ApplicationDataType.
implementationArrayElement	ImplementationDataTypeElement	0..1	ref	Reference to an ImplementationDataTypeElement in an array. This reference shall only be used if the referenced context element (VariableDataPrototype that is referenced by the SenderReceiverToSignalGroupMapping or ArgumentDataPrototype that is referenced by the ClientServerCompositeTypeMapping) is typed by an ImplementationDataType.
index	Integer	1	attr	Position of an element in an array.

Table 4.28: IndexedArrayElement

Class	**ClientServerRecordElementMapping**			
Package	M2::AUTOSARTemplates::SystemTemplate::DataMapping			
Note	Mapping of a primitive record element to a SystemSignal. If the ArgumentDataPrototype that is referenced by ClientServerCompositeTypeMapping is typed by an ApplicationDataType the reference to the ApplicationRecordElement shall be used. If the VariableDataPrototype is typed by the ImplementationDataType the reference to the ImplementationRecordElement shall be used. If the element is composite, there will be no mapping (multiplicity 0). In this case the "RecordElementMapping" Element will aggregate the "TypeMapping" Element. In that way also the composite datatypes can be mapped to SystemSignals. Regardless whether composite or primitive record element is mapped the record element always needs to be specified.			
Base	ARObject			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**
applicationRecordElement	ApplicationRecordElement	0..1	ref	Reference to a applicationRecordElement in the context of the mappedOperation or in the context of a composite element. This reference shall only be used if the ArgumentDataPrototype that is referenced by the ClientServerCompositeTypeMapping is typed by an ApplicationDataType.
complexTypeMapping	ClientServerCompositeTypeMapping	0..1	aggr	This aggregation will be used if the element is composite.
implementationRecordElement	ImplementationDataTypeElement	0..1	ref	Reference to a ImplementationRecordElement in the context of the mappedOperation or in the context of a composite element. This reference shall only be used if the ArgumentDataPrototype that is referenced by the ClientServerCompositeTypeMapping is typed by an ImplementationDataType.
systemSignal	SystemSignal	0..1	ref	Reference to the system signal used to carry the primitive ApplicationRecordElement.

Table 4.29: ClientServerRecordElementMapping

4.2.2 Signal Path Constraint

One of the tasks of the System Generator is actually to calculate automatically the communication (signals) between the RTEs and define the needed frames for that communication. These definitions of the frames include implicitly the definition of the paths the AUTOSAR-Signals are transmitted through the system. Thereby the System Generator often has the choice between alternative ways through the system. In the example shown in Figure 4.11 the System Generator would have the choice between two ways (Path1: CAN3 or Path2: CAN1-GW-CAN2) for a signal from ECU2 to ECU4. If no further information is given the decision will be made e.g. by means of boundary conditions like busload, transmissions speed, etc.

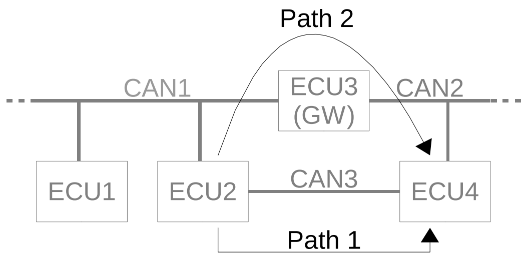

Figure 4.11: Example for a Communication Path

Signal Mapping Constraints allow to further restrict or specify the path(s) a signal is allowed to be transmitted over. A path is specified by an list of `PhysicalChannels`.

There exist four different constraints for signals regarding the signal path (see Figure 4.12):

1. The `CommonSignalPath` describes that two signals must take the same way (Signal Path) in the topology.
2. The `ForbiddenSignalPath` describes the way (Signal Path) that a signal must not take in the topology, e.g. in case of safety critical transmission.
3. The `PermissibleSignalPath` describes the way (Signal Path) a signal can take in the topology. If more than one PermissibleSignalPath is defined for the same signal/operation attributes, any of them can be chosen.
4. The `SeparateSignalPath` describes that two or more signals must not take the same way (Signal Path) in the topology e.g. in case of redundant transmission. It is also possible that the same signal is aggregated two times by the `SeparateSignalPath` element to indicate that this signal should be transmitted redundantly over two different paths.

The meta-model part, which describes the Communication Path constraints, will be explained in the following sections.

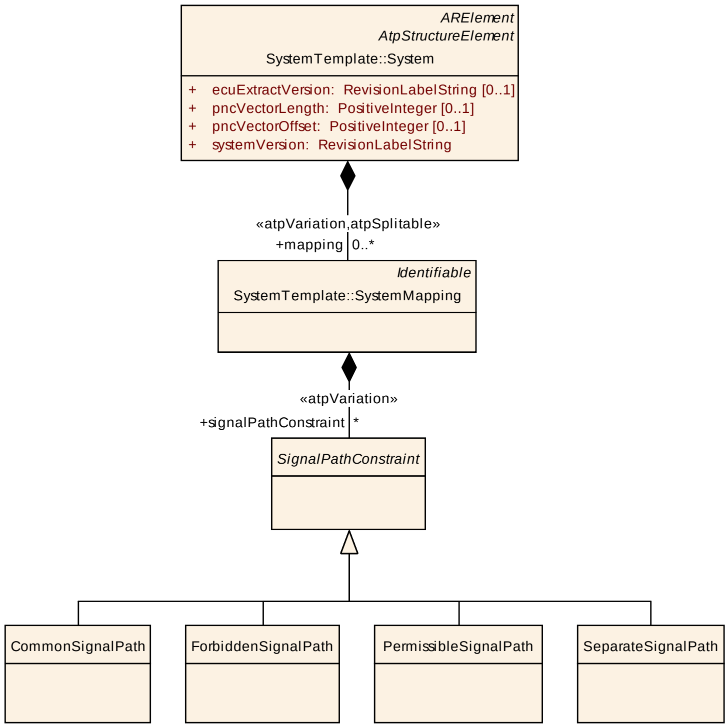

Figure 4.12: Communication Path Description (SignalPathConstraints)

4.2.2.1 CommonSignalPath

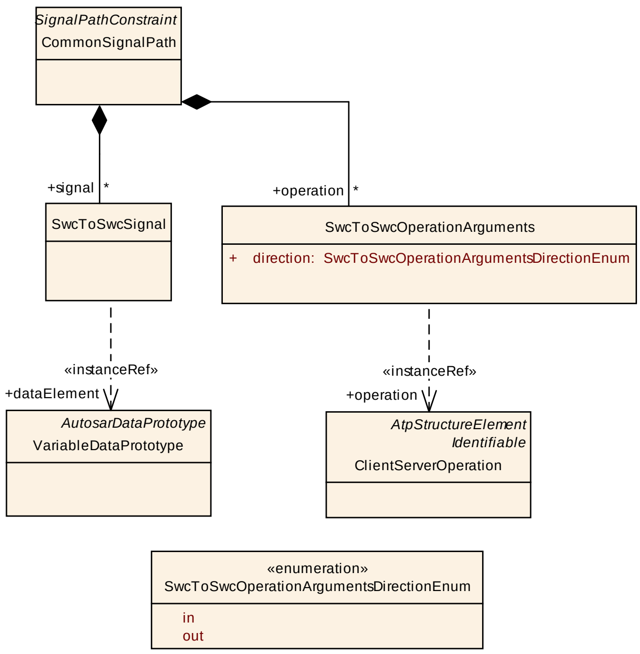

Figure 4.13: Description of signals that must take the same way in the topology (CommonSignalPath)

Class	**CommonSignalPath**			
Package	M2::AUTOSARTemplates::SystemTemplate::SignalPaths			
Note	The CommonSignalPath describes that two or more SwcToSwcSignals and/or SwcToSwcOperationArguments must take the same way (Signal Path) in the topology.			
Base	ARObject,SignalPathConstraint			
Attribute	*Datatype*	*Mul.*	*Kind*	*Note*
operation	SwcToSwcOperationArguments	*	aggr	
signal	SwcToSwcSignal	*	aggr	The SwcToSwcSignals that must take the same way (Signal Path) in the topology.

Table 4.30: CommonSignalPath

Class	**SwcToSwcSignal**			
Package	M2::AUTOSARTemplates::SystemTemplate::SignalPaths			
Note	The SwcToSwcSignal describes the information (data element) that is exchanged between two SW Components. On the SWC Level it is possible that a SW Component sends one data element from one P-Port to two different SW Components (1:n Communication). The SwcToSwcSignal describes exactly the information which is exchanged between one P-Port of a SW Component and one R-Port of another SW Component.			
Base	ARObject			
Attribute	*Datatype*	*Mul.*	*Kind*	*Note*
dataElement	VariableDataPrototype	2	iref	Reference to a data element on the PPortPrototype and to the same data element on the RPortPrototype.

Table 4.31: SwcToSwcSignal

Class	**SwcToSwcOperationArguments**			
Package	M2::AUTOSARTemplates::SystemTemplate::SignalPaths			
Note	The SwcToSwcOperationArguments describes the information (client server operation arguments, plus the operation identification, if required) that are exchanged between two SW Components from exactly one client to one server, or from one server back to one client. The direction attribute defines which direction is described. If direction == IN, all arguments sent from the client to the server are described by the SwcToSwcOperationArguments, in direction == OUT, it's the arguments sent back from server to client.			
Base	ARObject			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
direction	SwcToSwcOperationArgumentsDirectionEnum	1	attr	Direction addressed by this SwcToSwcClientServerOperation element.
operation	ClientServerOperation	2	iref	Reference to the operation at the client and at the server side whose arguments are described by SwcToSwcOperationArguments. The two ports referenced must be connected by a connector in the software component description.

Table 4.32: SwcToSwcOperationArguments

Enumeration	**SwcToSwcOperationArgumentsDirectionEnum**
Package	M2::AUTOSARTemplates::SystemTemplate::SignalPaths
Note	Direction addressed by this element.
Literal	***Description***
in	IN (all IN and INOUT arguments)
out	OUT (all OUT and INOUT arguments) .

Table 4.33: SwcToSwcOperationArgumentsDirectionEnum

4.2.2.2 ForbiddenSignalPath

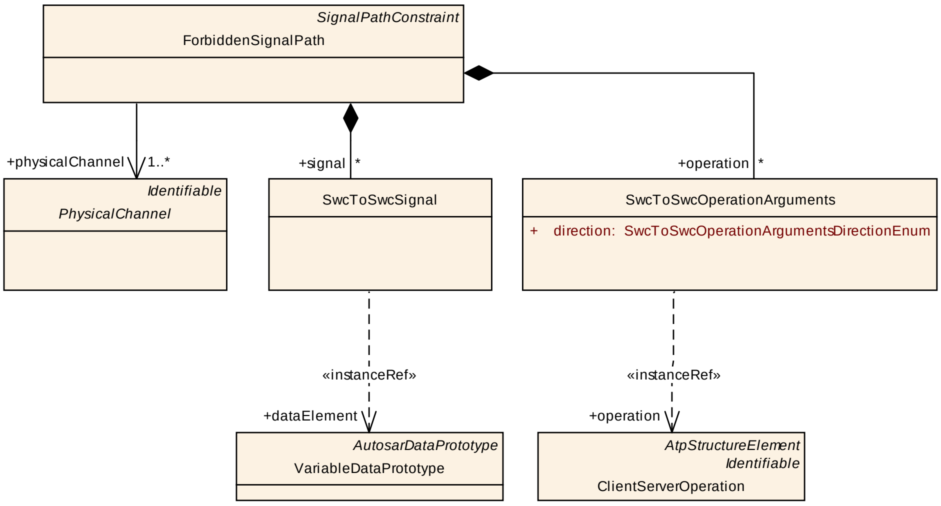

Figure 4.14: Description of the signal path that a signal must not take in the topology (ForbiddenSignalPath)

Class	**ForbiddenSignalPath**			
Package	M2::AUTOSARTemplates::SystemTemplate::SignalPaths			
Note	The ForbiddenSignalPath describes the physical channels which an element must not take in the topology. Such a signal path can be a constraint for the communication matrix, because such a path has an effect on the frame generation and the frame path.			
Base	ARObject,SignalPathConstraint			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
operation	SwcToSwcOperationArguments	*	aggr	Reference to the operation arguments of one operation which must not take the predefined way in the topology.
physicalChannel	PhysicalChannel	1..*	ref	The SwcToSwcSignal must not be transmitted on one of these physical channels.
signal	SwcToSwcSignal	*	aggr	The data element which must not take the predefined way in the topology.

Table 4.34: ForbiddenSignalPath

— AUTOSAR CONFIDENTIAL —

4.2.2.3 PermissibleSignalPath

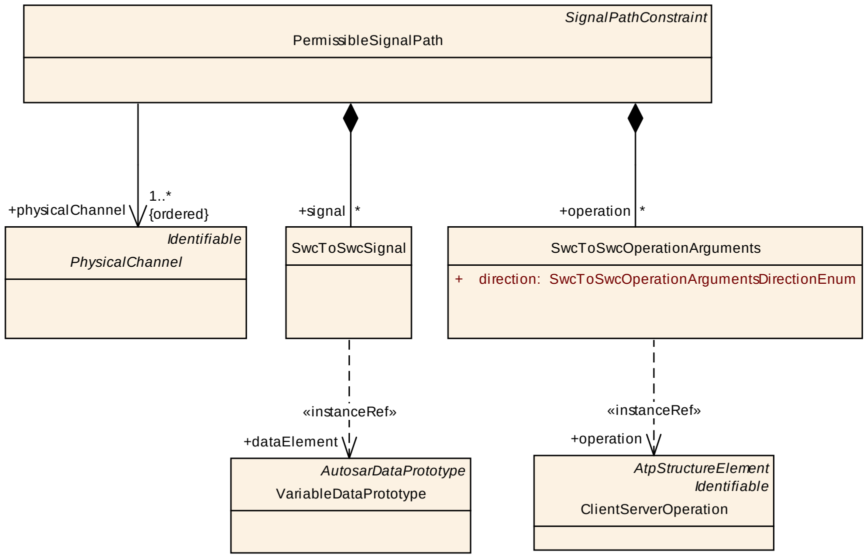

Figure 4.15: Description of the signal path that a signal must take in the topology (PermissibleSignalPath)

Class	**PermissibleSignalPath**			
Package	M2::AUTOSARTemplates::SystemTemplate::SignalPaths			
Note	The PermissibleSignalPath describes the way a data element shall take in the topology. The path is described by ordered references to PhysicalChannels. If more than one PermissibleSignalPath is defined for the same signal/operation attributes, any of them can be chosen. Such a signal path can be a constraint for the communication matrix . This path describes that one data element should take path A (e.g. 1. CAN channel, 2. LIN channel) and not path B (1. CAN channel, FlexRay channel A). This has an effect on the frame generation and the frame path.			
Base	ARObject,SignalPathConstraint			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
operation	SwcToSwcOperationArguments	*	aggr	The arguments of an operation that can take the predefined way in the topology.
physicalChannel (ordered)	PhysicalChannel	1..*	ref	The SwcToSwcSignal can be transmitted on one of these physical channels.
signal	SwcToSwcSignal	*	aggr	The data element which can take the predefined way in the topology.

Table 4.35: PermissibleSignalPath

4.2.2.4 SeparateSignalPath

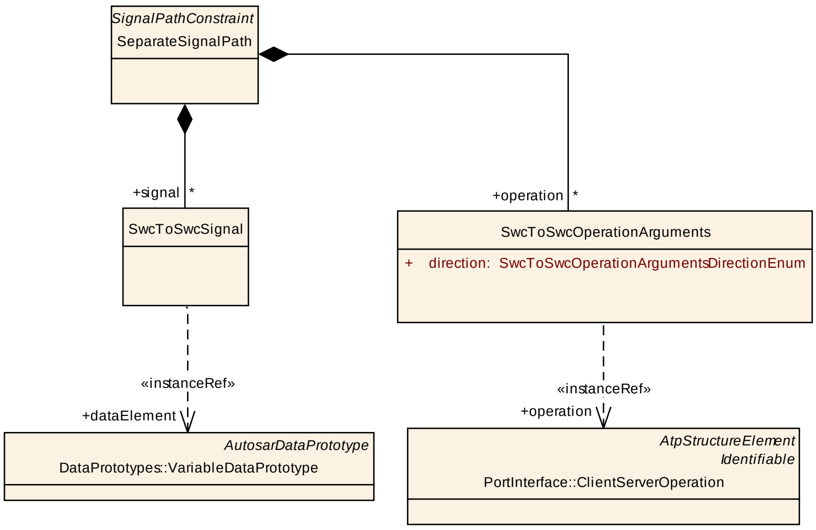

Figure 4.16: Description of signals that must not take the same way in the topology (SeparateSignalPath)

Class	**SeparateSignalPath**			
Package	M2::AUTOSARTemplates::SystemTemplate::SignalPaths			
Note	The SeparateSignalPath describes that two SwcToSwcSignals and/or SwcToSwcOperationArguments must not take the same way (Signal Path) in the topology (e.g. Redundancy). This means that the signals are not allowed to share even a single physical channel in their path.			
Base	ARObject,SignalPathConstraint			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
operation	SwcToSwcOperationArguments	*	aggr	The SwcToSwcOperationArguments that must not take the same way (Signal Path) in the topology.
signal	SwcToSwcSignal	*	aggr	The SwcToSwcSignals that must not take the same way (Signal Path) in the topology.

Table 4.36: SeparateSignalPath

4.3 RTE and basic software resource estimations

Important constraints for system partitioning are the available resources on the ECUs in the system. For SW components, the resource estimations can be stated in SW component descriptions. It is however not only SW components that require resources. AUTOSAR RTE and basic software running on the ECU have resource needs as well.

The realization of the RTE and the kind of basic software to be run on a certain ECU depend on the implicit and explicit usage of all basic software by the software components. The software components need to communicate internally and with software components on other ECUs. Furthermore, they have different needs with respect to scheduling. This results in implicit use of e.g. communication and operating system software. In addition, the software components make explicit use of basic software when they e.g. utilize system services (e.g. diagnostics) and access sensors/actuators via the I/O abstraction layer or the complex device driver abstraction layer. Thus, the resource consumption of the RTE and the basic software depend on the SW Components mapped to the ECU, since this determines the exact configuration of the RTE and the basic software.

The resource consumption for RTE and basic software are specified using class `EcuResourceEstimation`. Each estimation is performed for a specific ECU and for a specific set of SW mapped to that ECU (reference from `EcuResourceEstimation` to `ECUInstance` and `SwCompToEcuMapping`). Different resource estimations for a specific ECU, but with different mappings may exist, e.g. for different variants of the system, or to show the difference of resource needs for different mappings. The `EcuResourceEstimation` aggregates the meta-class `ResourceConsumption` from the GenericStructure package each for RTE and basic software, which specifies stack and heap usage and execution time.

ExecutionTime and StackUsage are used to provide information on the implementation specific resource usage of the `ExecutableEntity` defined in the `InternalBehavior` of SW-Component respectively in the BswBehavior of BSW Module. `MemorySection` documents the resources needed to load the object file containing the implementation on the ECU. `HeapUsage` describes the dynamic memory usage of the software.

Figure 4.17 shows the meta-model for resource estimations for RTE and basic SW.

— AUTOSAR CONFIDENTIAL —

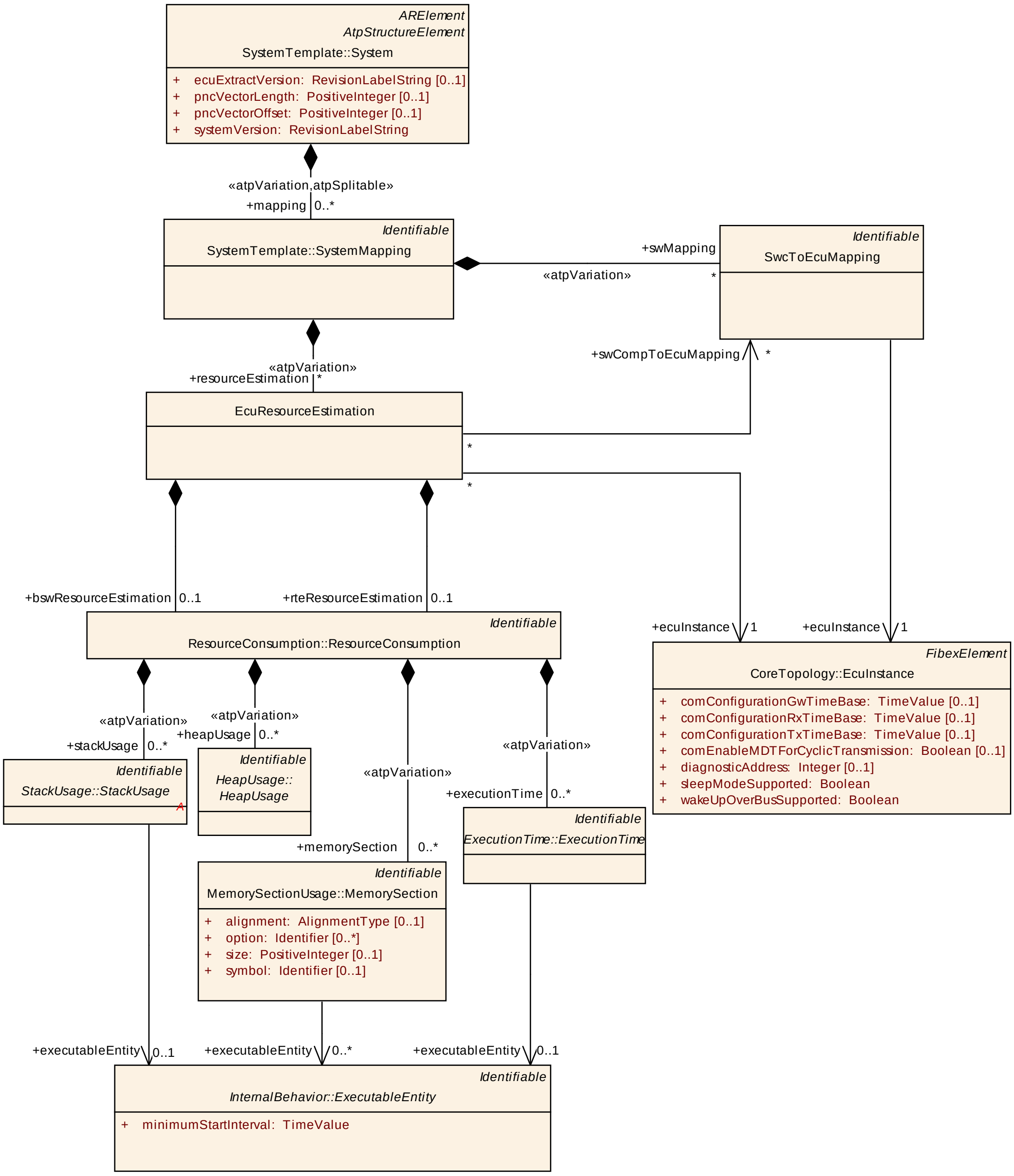

Figure 4.17: ECU resource estimations (ResourceEstimation)

[constr_3005] valid `EcuResourceEstimation` ⌈The same EcuInstance shall be referenced directly from the `EcuResourceEstimation` and from the `SwCompToEcuMapping`:

EcuResourceEstimation.swCompToEcuMapping.ecuInstance == EcuResourceEstimation.ecuInstance ⌋

Class	**EcuResourceEstimation**			
Package	M2::AUTOSARTemplates::SystemTemplate::SWmapping			
Note	Resource estimations for RTE and BSW of a single ECU instance.			
Base	ARObject			
Attribute	*Datatype*	*Mul.*	*Kind*	*Note*
bswResourceEstimation	ResourceConsumption	0..1	aggr	Estimation for the resource consumption of the basic software.
ecuInstance	EcuInstance	1	ref	Reference to the ECU this estimation is done for.
introduction	DocumentationBlock	0..1	aggr	This represents introductory documentation about the ecu resource estimation **Tags:** xml.sequenceOffset=-10
rteResourceEstimation	ResourceConsumption	0..1	aggr	Estimation for the resource consumption of the run time environment.
swCompToEcuMapping	SwcToEcuMapping	*	ref	References to SwCompToEcuMappings that have been taken into account for the resource estimations. This way it is possible to define dfferent EcuResourceEstimations with different mappings, e.g. before and after mapping an additional SW component.

Table 4.37: EcuResourceEstimation

Class	**ResourceConsumption**			
Package	M2::AUTOSARTemplates::CommonStructure::ResourceConsumption			
Note	Description of consumed resources by one implementation of a software.			
Base	ARObject,Identifiable,MultilanguageReferrable,Referrable			
Attribute	*Datatype*	*Mul.*	*Kind*	*Note*
executionTime	ExecutionTime	*	aggr	Collection of the execution time descriptions for this implementation. The aggregation of executionTime is subject to variability with the purpose to support the conditional existence of runnable entities. **Stereotypes:** atpVariation **Tags:** Vh.latestBindingTime=PreCompileTime
heapUsage	HeapUsage	*	aggr	Collection of the heap memory allocated by this implementation. **Stereotypes:** atpVariation **Tags:** Vh.latestBindingTime=PreCompileTime
memorySection	MemorySection	*	aggr	An abstract memory section required by this Implementation. **Stereotypes:** atpVariation **Tags:** Vh.latestBindingTime=PreCompileTime

— AUTOSAR CONFIDENTIAL —

Attribute	Datatype	Mul.	Kind	Note
sectionNamePrefix	SectionNamePrefix	*	aggr	A prefix to be used for the memory section symbol in the code. **Stereotypes:** atpVariation **Tags:** Vh.latestBindingTime=PreCompileTime
stackUsage	StackUsage	*	aggr	Collection of the stack memory usage for each runnable entity of this implementation. The aggregation of StackUsage is subject to variability with the purpose to support the conditional existence of runnable entities. **Stereotypes:** atpVariation **Tags:** Vh.latestBindingTime=PreCompileTime

Table 4.38: ResourceConsumption

The element ResourceConsumption and the subelements heapUsage, MemorySection, stackUsage and ExecutionTime are described in more detail in the BSW Module Description [16].

4.4 Partial Networking

The AUTOSAR BSW stack supports power saving during vehicle operation time with the partial networking mechanism. This mechanism allows to shut down and startup the bus communication interfaces of groups of ECUs (Partial Network Cluster) during normal bus communication.

On the VFB Level Partial Networks are represented by Virtual Function Clusters and are described with PortGroups. The Virtual Function Cluster groups the communication necessary to realize one or more vehicle functions that can become activated/deactivated during normal vehicle operation. Virtual Function Clusters are described in more detail in [4].

In the system description the Virtual Function Clusters are mapped onto Partial Network Clusters that are realized with IPduGroups.

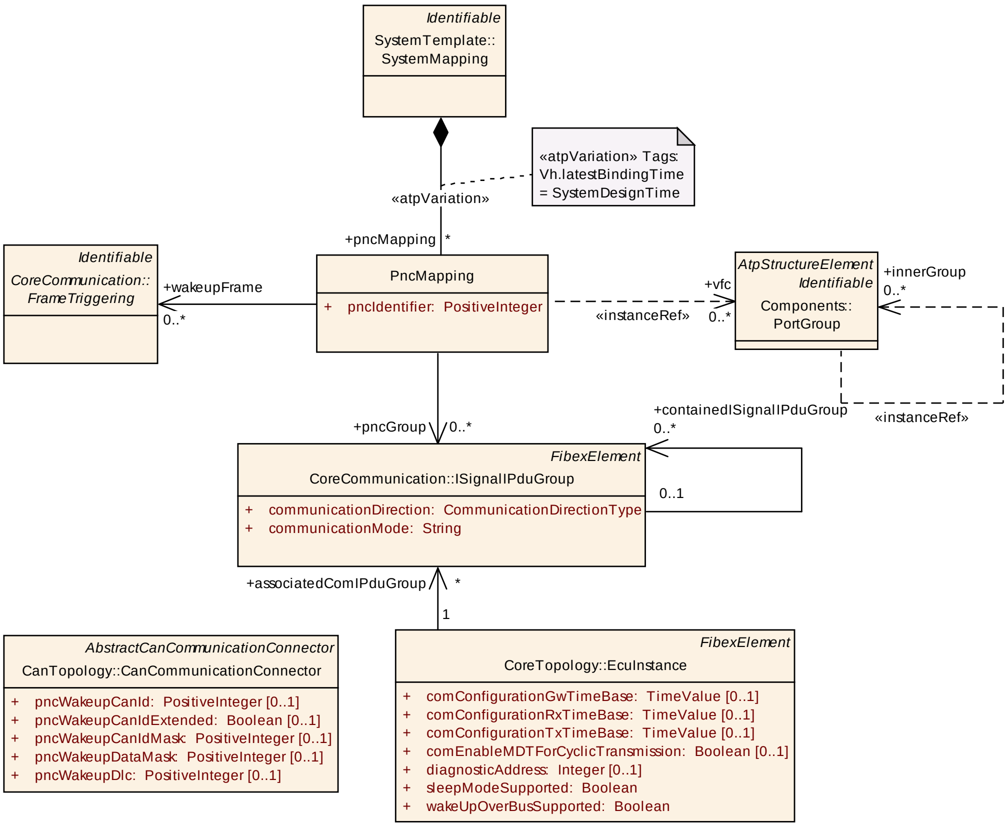

Figure 4.18: Mapping of Virtual Function Clusters onto Partial Network Clusters

Class	**PncMapping**			
Package	M2::AUTOSARTemplates::SystemTemplate::PncMapping			
Note	Describes a mapping between one or several Virtual Function Clusters onto Partial Network Clusters. A Virtual Function Cluster is realized by a PortGroup. A Partial Network Cluster is realized by one or more IPduGroups.			
Base	ARObject			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
pncGroup	ISignalIPduGroup	*	ref	IPduGroup participating in a Partial Network Cluster. This reference is optional in case an ecu extract has only indirect pnc access, i.e. ecu is not directly conntected to a network which supports partial network.
pncIdentifier	PositiveInteger	1	attr	Identifer of the Partial Network Cluster.
vfc	PortGroup	*	iref	Virtual Function Cluster to be mapped onto a Partial Network Cluster. This reference is optional in case that the System Description doesn't use a complete Software Component Description (VFB View). This supports the inclusion of legacy systems.

Attribute	Datatype	Mul.	Kind	Note
wakeupFrame	FrameTriggering	*	ref	Reference to collection of FrameTriggerings that are used for the wakeup of this PNC (Application Frames or Nm Frames can be used). This reference is optional in case an ecu extract has only indirect pnc access, i.e. ecu is not directly conntected to a network which supports partial network.

Table 4.39: PncMapping

The runtime information that is used to coordinate the request/release information of all partial networks is called `pncVector`. The size and position of the `pncVector` inside the network management user data is globally defined in the `System` class in chapter 1.4.

Attributes used to configure the Partial Network Wakeup of one specific Ecu are described in chapter 2.3.1.4.

5 Communication

This chapter describes all topics that deal with constraints or configurations that describe the information exchange between the ECUs. The description of communication matrices in the System Template is based on the description in ASAM FIBEX 3.0 [7]. Because of the requirements of AUTOSAR some extensions were made to the original FIBEX model.

The main elements to describe communication in the System Template are the `Signals` (System Signals and ISignals), `PDUs` (I-Pdus, N-Pdus and NmPdus) and `Frames`, as it can be seen on Figure 5.1.

A `Frame` is a piece of information that is exchanged over the communication channels. It has a payload section of a certain length in bytes, which contains an arbitrary number of non-overlapping `PDUs` (I-Pdus, N-Pdus, XCPPdus or NmPdus). In AUTOSAR only FlexRay supports the packing and unpacking of multiple PDUs into/out of one FlexRay Frame. The AUTOSAR CanIf and LinIf are not capable of packing multiple PDUs into one Frame. CAN Frames and LIN Frames shall contain only one Pdu.

A `PDU (Protocol Data Unit)` is the information delivered through a network layer. For the network to understand which layer is being discussed, a single-letter prefix is added to the PDU.

- I-PDU - Interaction Layer Protocol Data Unit (assembled and disassembled in COM) In the case of external communication the Interaction Layer packs one or more signals into assigned `I-Pdus` and passes them to the underlying layer for transfer between nodes in a network. The `I-Pdu` is described in the System Template by the `IPdu` element.
- N-PDU - Network Layer Protocol Data Unit (assembled and disassembled in a Transport Protocol module). The TP module′s main purpose is the segmentation and reassembly of `I-PDUs` that do not fit in one of the assigned `N-PDUs`. The `N-Pdu` is described in the System Template by the `NPdu` element.
- L-PDU - Data Link Layer Protocol Data Unit (assembled and disassembled in AUTOSAR Hardware Abstraction layer). The element `Frame` in the System Template represents the Autosar Layered Architectures `L-Sdu`. `Sdu` is the abbreviation of "Service Data Unit". The Data Link Layers `L-Pdu` contains the `L-Sdu` and `PCI` (Protocol Control Information). The L-PDU is not described in the System Template.

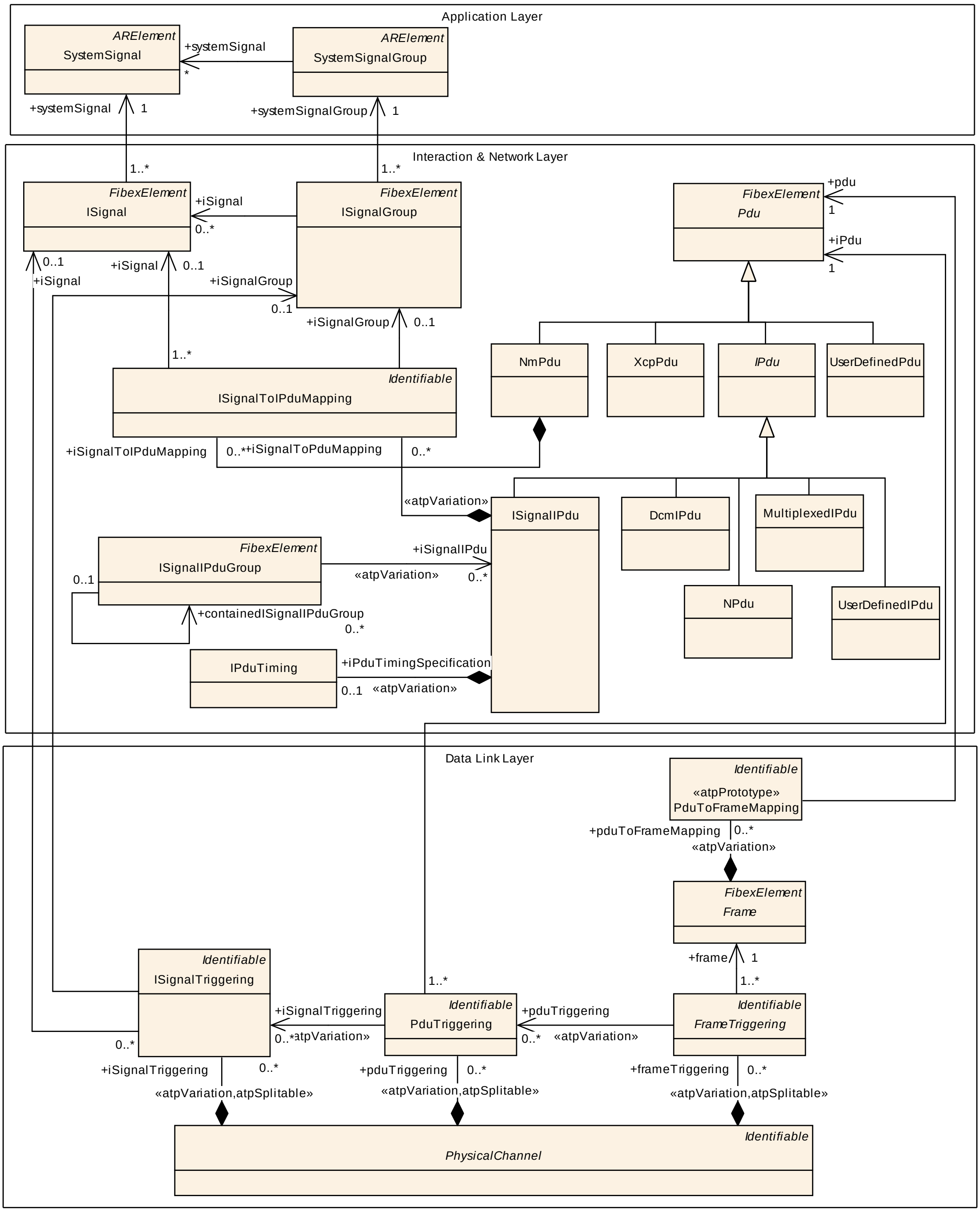

Figure 5.1: Communication Overview (FibexCore: Communication)

In case no multiplexing is performed the `I-PDUs` of COM that fit into one frame are passed directly via the PDU Router to the communication interfaces. For CAN and LIN the maximum L-PDU length is 8 bytes. For FlexRay the maximum L-PDU length is 254 bytes. Large `I-PDUs` that are too long to fit into one L-Pdu and `I-PDUs` which contain dynamic signals are routed via the Transport Layer to the communication interfaces.

The Transport Protocols are described in more detail in chapter 5.15. All I-PDUs from the DCM are transported via the Transport Protocol. [1]

If multiplexing is performed an IPdu is routed between the IPdu Multiplexer and the Interface Layer or Transport Layer. To distinguish this two different cases two specializations `SignalIPdu` and `MultiplexedIPdu` are introduced. A `SignalIPdu` represents an I-PDU handled by Com. A `MultiplexedIPdu` describes the combination of Signal IPdu's performed by the multiplexer, to be sent or received between the multiplexer and the interfaces. The Multiplexer is described in more detail in chapter 5.6.

AUTOSAR COM provides the possibility to define Transmission Modes for each COM IPdu. For this reason the `SignalIPdu` aggregates the `IPduTiming`. The Transmission Modes are described in more detail in chapter 5.14.

5.1 Triggerings and Ports

The elements `FrameTriggering`, `PduTriggering` and `SignalTriggering` are describing the usage of Frames, IPdus and Signals on a physical channel.

A `FrameTriggering` need to fulfill requirements for contained Pdus that are defined by the corresponding `PduTriggerings`. And the `PduTriggering` need to fulfill requirements for contained `ISignals` that are defined by the corresponding `ISignalTriggerings`. The references between the Triggering elements can be used to describe these relationships. More details can be found in class tables of `FrameTriggering`, `PduTriggering` and `ISignalTriggering`.

In AUTOSAR the timing of bus messages can be controlled by send requests of the Application layer in combination with the COM Transmission Modes and Transfer Properties (esp. CAN). On the other hand it can be controlled by the FlexRay or LIN Interface. In this case the Bus Interface only requests I-PDUs that have to be provided by COM.

In the System Template the COM controlled timing is described with the aggregation between the `SignalIPdu` and the `IPduTiming`. The LIN and FlexRay Scheduling Tables are described in the `FrameTriggering`. Timing requirements for FlexRay, TTCAN and LIN Pdus can be specified with the Timing Extension model. More details are described in chapter 1.4.4.

[1]There is one special gateway use case where a Transport Protocol NPdu can be routed directly by the Pdu Router and where the TP module is not involved. More details can be found in chapter 5.15.

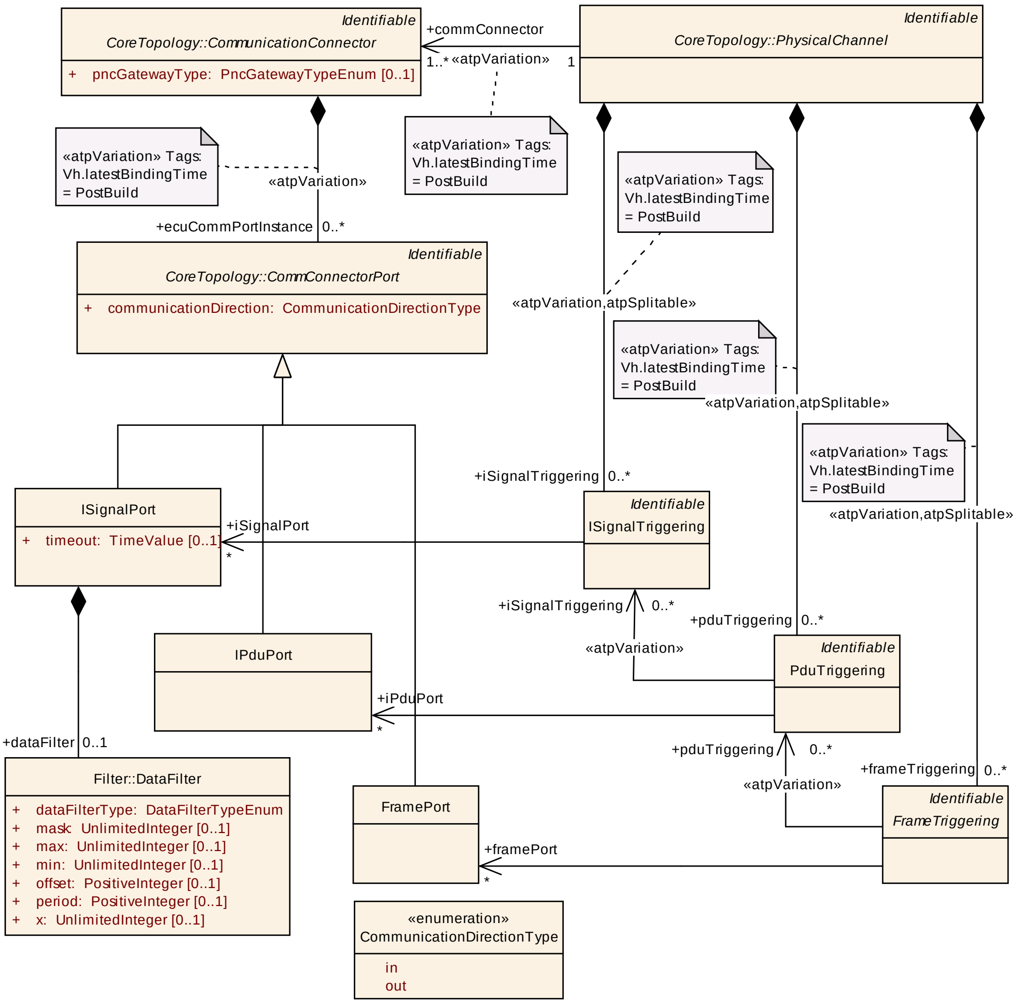

Figure 5.2: Communication Matrix (FibexCore: CommunicationMatrix)

Figure 5.2 shows the relationship between the `CommConnectorPort` and the `FrameTriggering`, `PduTriggering` and `SignalTriggering`. This relationship allows to specify explicitly which frames, IPdus, Signals are received/sent by the connected ECU on the connected channel.

The following rules apply for the creation of Pdu Triggering and Pdu Ports:

- `UserDefinedPdus`, `NmPdus`, `NPdus` which are not going through the Pdu Router get their triggering information via the containing `FrameTriggering` and `FramePort` (no Pdu Triggering is defined for these Pdus).
- In case of a low level routing of `NPdus` the Pdus are handled like `IPdus` and the `PduTriggering` and `IPduPort` shall be defined.
- `DcmIPdus` shall have `PduTriggering` and `IPduPorts` since they are handled by the PduR (connection to the Dcm and/or DcmIPdu-routing).
- `SignalIpdus` that are part of a `MultiplexedIpdu` (static or dynamic) and are also handled by the Com module shall have `PduTriggering` and `IPduPorts`

since they are handled by the PduR (and Com). Especially it is allowed to ignore certain received parts of a MultiplexedIpdu in a specific ECU.

- SignalIPdus (not part of MultiplexedIPdus), UserDefinedIPdus and MultiplexedIPdus shall have a PduTriggering and IPduPort if they are handled by the PduR. Especially it is allowed to ignore a certain IPdu out of a Flexray frame if it is not considered in a specific ECU.
- In case a NmPdu contains user data and is handled by the BusNm via the PduR and Com the NmPdu gets PduTriggering and IPduPort.

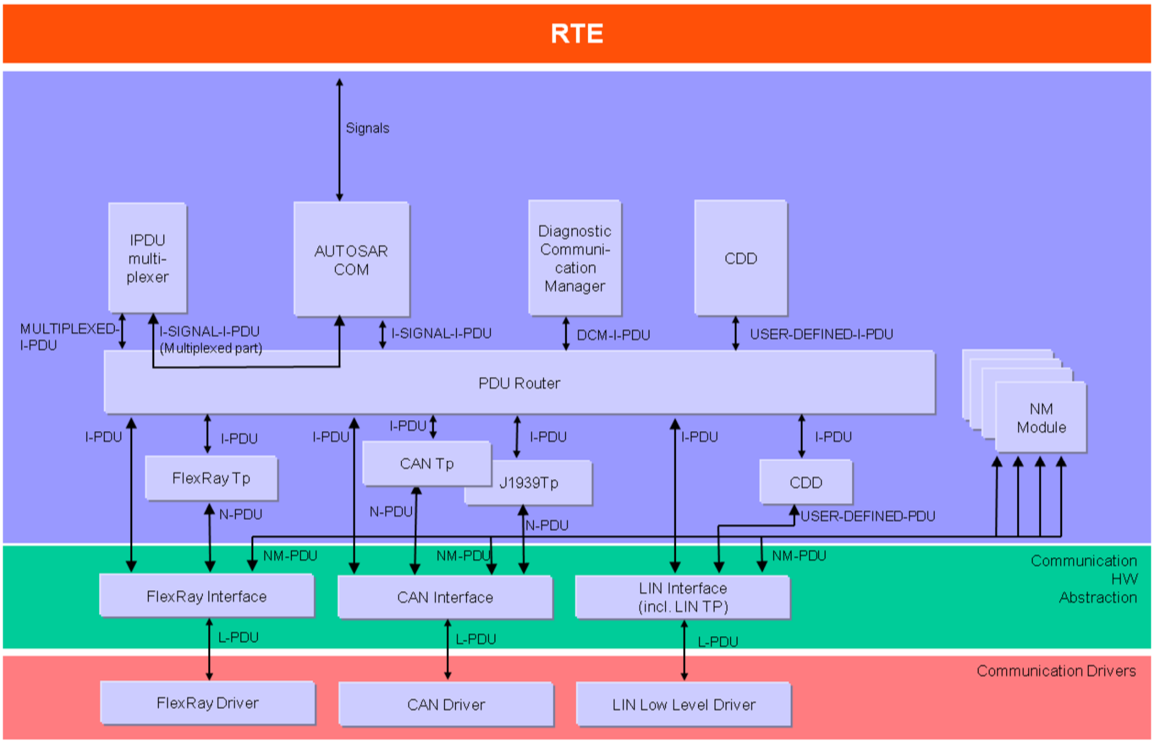

Figure 5.3: AUTOSAR Layered Architecture

Class	CommConnectorPort (abstract)			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::FibexCore::CoreTopology			
Note	The Ecu communication relationship defines which signals, Pdus and frames are actually received and transmitted by this ECU. For each signal, Pdu or Frame that is transmitted or received and used by the Ecu an association between an ISignalPort, IPduPort or FramePort with the corresponding Triggering shall be created. An ISignalPort shall be created only if the corresponding signal is handled by COM (RTE or Signal Gateway). If a Pdu Gateway ECU only routes the Pdu without being interested in the content only a FramePort and an IPduPort needs to be created.			
Base	ARObject,Identifiable,MultilanguageReferrable,Referrable			
Attribute	Datatype	Mul.	Kind	Note
communicationDirection	CommunicationDirectionType	1	attr	Communication Direction of the Connector Port (input or output Port).

Table 5.1: CommConnectorPort

Class	FramePort			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::FibexCore::CoreCommunication			
Note	Connectors reception or send port on the referenced channel referenced by a FrameTriggering.			
Base	ARObject,CommConnectorPort,Identifiable,MultilanguageReferrable,Referrable			
Attribute	Datatype	Mul.	Kind	Note
–	–	–	–	–

Table 5.2: FramePort

Class	IPduPort			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::FibexCore::CoreCommunication			
Note	Connectors reception or send port on the referenced channel referenced by a PduTriggering.			
Base	ARObject,CommConnectorPort,Identifiable,MultilanguageReferrable,Referrable			
Attribute	Datatype	Mul.	Kind	Note
–	–	–	–	–

Table 5.3: IPduPort

Class	ISignalPort			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::FibexCore::CoreCommunication			
Note	Connectors reception or send port on the referenced channel referenced by an ISignalTriggering. If different timeouts or DataFilters for ISignals need to be specified several ISignalPorts may be created.			
Base	ARObject,CommConnectorPort,Identifiable,MultilanguageReferrable,Referrable			
Attribute	Datatype	Mul.	Kind	Note

Attribute	Datatype	Mul.	Kind	Note
dataFilter	DataFilter	0..1	aggr	Optional specification of a signal COM filter at the receiver side in case that the System Description doesn't use a complete Software Component Description (VFB View). This supports the inclusion of legacy system signals. If a full DataMapping exist for the SystemSignal this information may be available from a configured ReceiverComSpec. In this case the ReceiverComSpec overrides this optional specification.
timeout	TimeValue	0..1	attr	Optional timeout value in seconds for the reception of the ISignal. In case the System Description doesn't use a complete Software Component Description (VFB View). This supports the inclusion of legacy system signals. If a full DataMapping exist for the SystemSignal this information may be available from a configured ReceiverComSpec, in this case the timeout value in ReceiverComSpec override this optional timeout specification.

Table 5.4: ISignalPort

5.1.1 Port elements in System Extract/ECU Extract

The processing in the ECU determine the existence of ports in the System Extract/Ecu Extract. In case that a Gateway ECU only routes a `Frame` without being interested in the content leads to a reduced description in the System Extract/ECU Extract. The following items describe the different scenarios and the consequences for the System Extract/ECU Extract description. A complete System Description contains all informations (scenario 1).

1) ECU that is sending or receiving a `Frame` and is interested in the content:

- One FramePort shall be used.
- One IPduPort shall be used.
- One ISignalPort is recommended. If different timeouts for signals need to be specified several ISignalPorts may be created.

The initial ECU Configuration Generator configures COM, PduR and lower layers with the information from the System Extract/ECU Extract.

2) Signal Gateway ECU that is sending or receiving a `Frame`:

- One FramePort shall be used.
- One IPduPort shall be used.

- One ISignalPort is recommended. If different timeouts for signals need to be specified several ISignalPorts may be created.

The initial ECU Configuration Generator configures COM, PduR and lower layers with the information from the System Extract/ECU Extract.

3) Pdu Gateway ECU that is sending or receiving a `Frame` (not interested in the content of the Pdu):

- One FramePort shall be used.
- One IPduPort shall be used.
- ISignalPorts shall not be created for this Gateway Ecu

— AUTOSAR CONFIDENTIAL —

5.2 Stereotype atpSplitable in the System Template

The stereotype ≪atpSplitable≫ is used in the System Template to support step-wise processes, where the System Configuration Description is completed incrementally over a development process. Example:

1) Description of Communication only consists of interaction signals (ISignal). This is enough information to create an individual ECU's RTE, and even contains enough information to configure an ECU where the actual Frame/PDU communication is being handled post-build.

2) In a second step, the communication matrix is being completed for a concrete vehicle. PDUs and Frames, along with their Triggerings are being added to the previous System Description. This model then contains the full information about an ECU's communication, especially containing the additional information to generate the post build information.

So, in this 2-step approach, an OEM could deliver the incomplete ECU extract from step (1) to the ECU integrator, who can then build a complete software image for the ECU. In the 2nd step, the ECU extract will be completed by the previously missing information, but as the first extract will still be valid due to the ≪atpSplitable≫ construct, the ECU including the flashed image from step (1) can be (re)used as it is, and just will be completed with the post build information, e.g. Frames and PDUs.

Further details about the ≪atpSplitable≫ stereotype can be found in the Generic Structure Template [1].

5.3 ISignals

SystemSignals can be defined independently of frames and communication clusters. The SystemSignals are unique per System and are representing the VariableDataPrototypes and ClientServerOperations in the communication description.

The RTE supports a "signal fan-out" where the same signal (System Signal) is sent in different IPdus to multiple receivers. The Pdu Router supports the "PDU fan-out" where the same IPdu is sent to multiple destinations.

To support the "signal fan-out" ISignals and ISignalGroups are introduced. An ISignal represents the Signal of the Interaction Layer. In the case of "signal fan-out", several ISignals in different IPdus refer to the same SystemSignal. The "Signal fan-out" must be executed by the RTE. ISignals describe the Interface between the precompile configured RTE and the potentially postbuild configured Com Stack.

The ISignalToIPduMapping element describes the mapping of ISignals to SignalIPdus and defines the position of an ISignal within a SignalIPdu.

[constr_3009] Overlapping of ISignals is prohibited ⌈ ISignals mapped to an ISignalIPdu MUST NOT overlap. ⌋

[constr_3010] ISignalIPdu length shall not be exceeded ⌈ The combined length of all ISignals and updateIndicationBits that are mapped into an ISignalIPdu shall not exceed the defined ISignalIPdu length. ⌋

[constr_3011] Overlapping of updateIndicationBits for ISignals is prohibited ⌈ The updateIndicationBitPosition for an ISignal in an ISignalIPdu MUST NOT overlap with other updateIndicationBitPositions and ISignal locations. ⌋

An ISignal aggregates the swDataDefProps element. With this aggregation the actual representation of the ISignal on the network can be specified. This representation follows a particular policy that is defined with the dataTypePolicy attribute.

For an alternative network representation it is important to define an alternative swDataDefProps especially CompuMethod defining alternative numerical representation and BaseType defining alternative encoding (e.g. from float in PortInterface to integer on bus). In case that the System Description doesn't use a complete Software Component Description (VFB View) the swDataDefProps are used to configure the Data Semantics.

The swDataDefProps element contains a reference to the SwBaseType. This reference can be used for the derivation of the ComSignalType in the COM Configuration. The ComSignalType shall be derived from the baseTypeSize (or maxBaseTypeSize) and the baseTypeEncoding.

The following table shows how the mapping onto the ComSignalType enumeration is done:

BaseTypeEncoding	*BaseTypeSize*	*ComSignalType*
2C	8 bits	SINT8
2C	16 bits	SINT16
2C	32 bits	SINT32
NONE	8 bits	UINT8
NONE	16 bits	UINT16
NONE	32 bits	UINT32
IEE754	32 bits	FLOAT32
IEE754	64 bits	FLOAT64
ISO-8859-1	baseTypeSize	UINT8_N (from the definition of baseTypeSize the ComSignalLength can be determined)
ISO-8859-2	baseTypeSize	UINT8_N (from the definition of baseTypeSize the ComSignalLength can be determined)
WINDOWS-1252	baseTypeSize	UINT8_N (from the definition of baseTypeSize the ComSignalLength can be determined)
UTF-8	baseTypeSize	UINT8_N (from the definition of baseTypeSize the ComSignalLength can be determined)
UCS-2	baseTypeSize	UINT8_N (from the definition of baseTypeSize the ComSignalLength can be determined)
-	maxBaseTypeSize	UINT8_DYN (from the definition of maxBaseTypeSize the ComSignalLength can be determined)
BOOLEAN	-	BOOLEAN

Table 5.5: SwBaseType to ComSignalType Mapping

The invalidValue is aggregated by the swDataDefProps element. The swDataDefProps and the swBaseType classes are described in more detail in the Software Component Template [4].

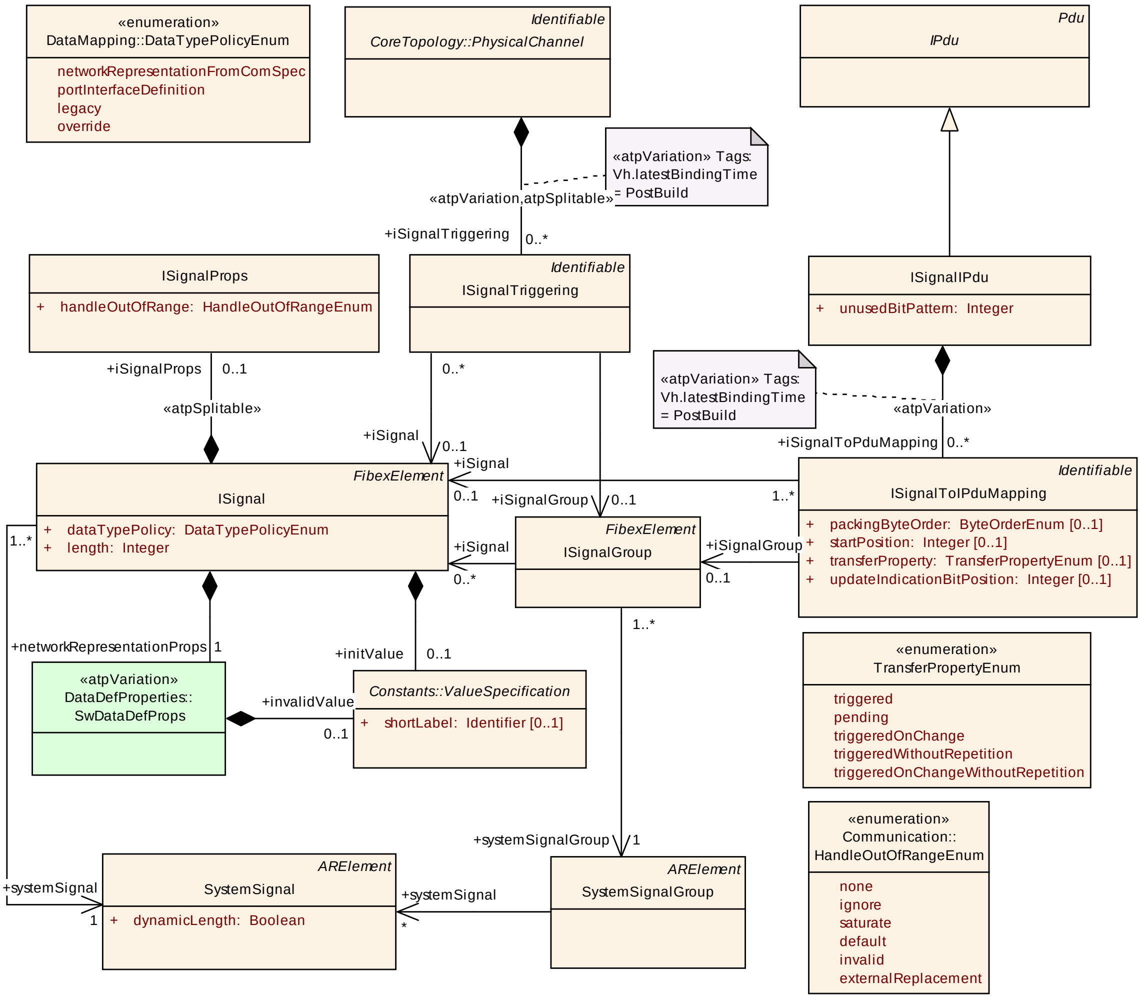

Figure 5.4: ISignals and the mapping into IPdus (FibexCore: SignalOverview)

The configuration of the COM Module for atomic signals can largely be derived from the System Template. A COM signal must be defined in the COM module configuration for each `ISignalToPduMapping` that is transmitted or received by the regarded ECU.

To support the AUTOSAR concept of complex data types the AUTOSAR COM layer provides signal groups. Every record or array element of a complex data type requires a `SystemSignal` for the transmission. But the RTE has to guarantee the consistent transmission of data. A signal group shall be transmitted and received consistently; therefore it provides data consistency for complex data types. A `SystemSignalGroup` refers to a set of `SystemSignals` that must always be kept together in a common IPdu. An `ISignalGroup` represents a Signal Group of the Interaction Layer. In the case of "signal fan-out", several `ISignalGroups` refer to the same `SystemSignalGroup`.

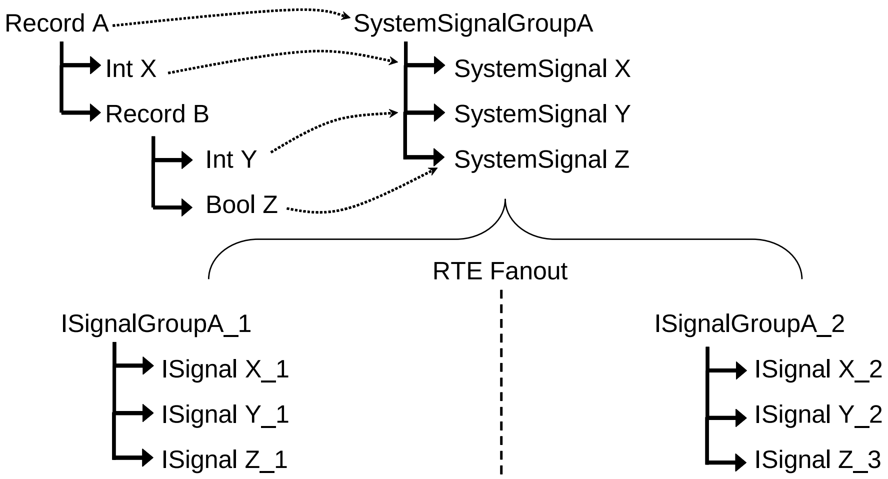

Figure 5.5: ISignal example

The example in Figure 5.5 shows the usage of `ISignalGroups` and `ISignals`. In this example a record is mapped to a `SystemSignalGroup`. All Application-RecordElements with primitive Datatypes are mapped to individual `SystemSignals`. If the same `SystemSignalGroup` is sent to different receivers (RTE Fanout) then two different `ISignalGroups` are created. For each `SystemSignal` within the `SystemSignalGroup` an `ISignal` is created. The different `ISignals` of the same `SystemSignal` can have different network representations.

Class	**ISignal**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::FibexCore::CoreCommunication			
Note	Signal of the Interaction Layer. The RTE supports a "signal fan-out" where the same System Signal is sent in different SignalIPdus to multiple receivers. The System Signal is unique per System. To support the RTE "signal fan-out" each SignalIPdu contains ISignals. If the same System Signal is to be mapped into several SignalIPdus there is one ISignal needed for each ISignalToIPduMapping. ISignals describe the Interface between the Precompile configured RTE and the potentially Postbuild configured Com Stack (see ECUC Parameter Mapping). In case of the SystemSignalGroup an ISignal must be created for each SystemSignal contained in the SystemSignalGroup. **Tags:** atp.recommendedPackage=ISignals			
Base	ARObject,CollectableElement,FibexElement,Identifiable,Multilanguage Referrable,PackageableElement,Referrable			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***

Attribute	*Datatype*	*Mul.*	*Kind*	*Note*
dataTypePolicy	DataTypePolicyEnum	1	attr	With the aggregation of SwDataDefProps an ISignal specifies how it is represented on the network. This representation follows a particular policy. Note that this causes some redundancy which is intended and can be used to support flexible development methodology as well as subsequent integrity checks. In particular, if the policy "portInterfaceDefinition" is chosen the network representation needs to be compatible to the default datatype specified in the interface. If the policy "networkRepresentationFromComSpec" is chosen the network representation needs to be compatible to the network representation of the ComSpec that is aggregated by the PortPrototype. If the "override" policy is chosen the requirements specified in the PortInterface and in the ComSpec are not fulfilled by the networkRepresentationProps. In case the System Description doesn't use a complete Software Component Description (VFB View) the "legacy" policy can be chosen.
iSignalProps	ISignalProps	0..1	aggr	Additional optional ISignal properties that may be stored in different files. **Stereotypes:** atpSplitable
initValue	ValueSpecification	0..1	aggr	Optional definition of a ISignal's initValue in case the System Description doesn't use a complete Software Component Description (VFB View). This supports the inclusion of legacy system signals. This value can be used to configure the Signal's "InitValue". If a full DataMapping exist for the SystemSignal this information may be available from a configured SenderComSpec and ReceiverComSpec. In this case the initvalues in SenderComSpec and/or ReceiverComSpec override this optional value specification. Further restrictions apply from the RTE specification.
length	Integer	1	attr	Size of the signal in bits. The size needs to be derived from the mapped VariableDataPrototype according to the mapping of primitive DataTypes to BaseTypes as used in the RTE. Indicates maximum size for dynamic length signals. The ISignal length of zero bits is allowed.

Attribute	Datatype	Mul.	Kind	Note
networkRepresentationProps	SwDataDefProps	1	aggr	Specification of the actual network representation. The usage of SwDataDefProps for this purpose is restricted to the attributes compuMethod and baseType. The optional baseType attributes "memAllignment" and "byteOrder" shall not be used. The attribute "dataTypePolicy" in the SystemTemplate element defines whether this network representation is compatible to the default datatype specified in the PortInterface or to the network representation of the comspec. If "override" is chosen by the system integrator the network representation can violate against the requirments defined in the PortInterface and in the network representation of the comspec. In case that the System Description doesn't use a complete Software Component Description (VFB View) this element is used to configure "ComSignalDataInvalidValue" and the Data Semantics. If a full DataMapping exist for the SystemSignal this information is additionally available from the mapped VariableDataElement. In this case the referenced datatypes needs to be compatible. Note that this redundancy is intended and can be used to support flexible development methodology as well as subsequent integrity checks.
systemSignal	SystemSignal	1	ref	Reference to the System Signal that is supposed to be transmitted in the ISignal.

Table 5.6: ISignal

Enumeration	**DataTypePolicyEnum**
Package	M2::AUTOSARTemplates::SystemTemplate::DataMapping
Note	This class lists the supported DataTypePolicies.
Literal	*Description*
legacy	In case the System Description doesn't use a complete Software Component Description (VFB View) this value can be chosen. This supports the inclusion of legacy signals. The aggregation of SwDataDefProps can be used to configure the "ComSignalDataInvalidValue" and the Data Semantics.
networkRepresentationFromComSpec	If this value is chosen the ISignal network representation must be compatible to the network representation of the comspec.

override	If this value is chosen the requirmenents specified in the PortInterface (portInterfaceDefinition) and in the comspec (networkRepresentationFromComSpec) are not fullfilled by the aggregated SwDataDefProps.
portInterface Definition	If this value is chosen the ISignal network representation must be compatible to the default datatype specified in the PortInterface (portInterfaceDefinition).

Table 5.7: DataTypePolicyEnum

Class	**ISignalProps**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::FibexCore::CoreCommunication			
Note	Additional ISignal properties that may be stored in different files.			
Base	ARObject			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
handleOut OfRange	HandleOutOfRa ngeEnum	1	attr	This attribute defines the outOfRangeHandling for received and sent signals.

Table 5.8: ISignalProps

Enumeration	**HandleOutOfRangeEnum**
Package	M2::AUTOSARTemplates::SWComponentTemplate::Communication
Note	A value of this type is taken for controlling the range checking behavior of the AUTOSAR RTE.
Literal	***Description***
default	The RTE will use the initValue if the actual value is out of the specified bounds.
external Replacement	This indicates that the value replacement is sourced from the externalReplacement.
ignore	The RTE will ignore any attempt to send or receive the corresponding dataElement if the value is out of the specified range.
invalid	The RTE will use the invalidValue if the value is out of the specified bounds.
none	A range check is not required.
saturate	The RTE will saturate the value of the dataElement such that it is limited to the applicable upper bound if it is greater than the upper bound. Consequently, it is limited to the applicable lower bound if the value is less than the lower bound.

Table 5.9: HandleOutOfRangeEnum

Class	ISignalGroup			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::FibexCore::CoreCommunication			
Note	SignalGroup of the Interaction Layer. The RTE supports a "signal fan-out" where the same System Signal Group is sent in different SignalIPdus to multiple receivers. An ISignalGroup refers to a set of ISignals that must always be kept together. A ISignalGroup represents a COM Signal Group. Therefore it is recommended to put the ISignalGroup in the same Package as ISignals (see atp.recommendedPackage) **Tags:** atp.recommendedPackage=ISignalGroup			
Base	ARObject,CollectableElement,FibexElement,Identifiable,Multilanguage Referrable,PackageableElement,Referrable			
Attribute	Datatype	Mul.	Kind	Note
iSignal	ISignal	*	ref	Reference to a set of ISignals that must always be kept together.
systemSignalGroup	SystemSignalGroup	1	ref	Reference to the SystemSignalGroup that is defined on VFB level and that is supposed to be transmitted in the ISignalGroup.

Table 5.10: ISignalGroup

Class	SystemSignalGroup			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::FibexCore::CoreCommunication			
Note	A signal group refers to a set of signals that must always be kept together. A signal group is used to guarantee the atomic transfer of AUTOSAR composite data types. The SystemSignalGroup defines a signal grouping on VFB level. On cluster level the Signal grouping is described by the ISignalGroup element. **Tags:** atp.recommendedPackage=SystemSignalGroups			
Base	ARElement,ARObject,CollectableElement,Identifiable,Multilanguage Referrable,PackageableElement,Referrable			
Attribute	Datatype	Mul.	Kind	Note
systemSignal	SystemSignal	*	ref	Reference to a set of SystemSignals that must always be kept together.

Table 5.11: SystemSignalGroup

Class	ISignalToIPduMapping			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::FibexCore::CoreCommunication			
Note	An ISignalToIPduMapping describes the mapping of ISignals to ISignalIPdus and defines the position of the ISignal within an ISignalIPdu. This element does NOT describe signal or IPdu fan-out but is used to describe the COM Signal Gateway fan-out. In case the ISignal/ISignalGroup is not part of the Signal Gateway the ISignal/ISignalGroup can only be mapped into one ISignalIPdu. In case the ISignal/ISignalGroup is part of the Signal Gateway several ISignalToIPduMappings of the same ISignal are supported.			
Base	ARObject,Identifiable,MultilanguageReferrable,Referrable			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**
iSignal	ISignal	0..1	ref	Reference to a ISignal that is mapped into the ISignalIPdu. Several ISignalToPduMappings to the same ISignal are only relevant when the ECU handles the signal gateway. Each ISignal contained in the ISignalGroup shall be mapped into an IPdu by an own ISignalToIPduMapping. The references to the ISignal and to the ISignalGroup in an ISignalToIPduMapping are mutually exclusive.
iSignalGroup	ISignalGroup	0..1	ref	Reference to an ISignalGroup that is mapped into the SignalIPdu. If an ISignalToIPduMapping for an ISignalGroup is defined, only the UpdateIndicationBitPosition and the transferProperty is relevant. The startPosition and the packingByteOrder shall be ignored. Each ISignal contained in the ISignalGroup shall be mapped into an IPdu by an own ISignalToIPduMapping. The references to the ISignal and to the ISignalGroup in an ISignalToIPduMapping are mutually exclusive.
packingByteOrder	ByteOrderEnum	0..1	attr	This parameter defines the order of the bytes of the signal and the packing into the SignalIPdu. The byte ordering "Little Endian" (MostSignificantByteLast), "Big Endian" (MostSignificantByteFirst) and "Opaque" can be selected. For opaque data endianness conversion shall be configured to Opaque. The value of this attribute impacts the absolute position of the signal into the SignalIPdu (see the startPosition attribute description). For an ISignalGroup the packingByteOrder is irrelevant and shall be ignored.

Attribute	Datatype	Mul.	Kind	Note
startPosition	Integer	0..1	attr	This parameter is necessary to describe the bitposition of a signal within an SignalIPdu. It denotes the least significant bit for "Little Endian" and the most significant bit for "Big Endian" packed signals within the IPdu (see the description of the packingByteOrder attribute). In AUTOSAR the bit counting is always set to "sawtooth" and the bit order is set to "Decreasing". Please note that the way the bytes will be actually sent on the bus does not impact this representation: they will always be seen by the software as a byte array. If a mapping for the ISignalGroup is defined, this attribute is irrelevant and shall be ignored.
transferProperty	TransferPropertyEnum	0..1	attr	The triggered or triggeredOnChange, triggeredWithoutRepetition and triggeredOnChangeWithoutRepetition transferProperty causes immediate transmission of the IPdu, except if transmission mode Periodic or transmission mode NONE is defined for the IPdu. The Pending transfer property does not cause transmission of an I-PDU. The immediate transmission of the IPdu is caused even if only one Signal of an IPdu has the transferProperty triggered or triggeredWithoutRepetition or triggeredOnChange or triggeredOnChangeWithoutRepetition and all other Signals have the transferProperty pending. Also for ISignals of an ISignalGroup (GroupSignals) this attribute is relevant and shall be evaluated: • If none of the ISignals belonging to the ISignalGroup have a transferProperty defined the transferProperty of the ISignalToPduMapping referring to the ISignalGroup is considered. • If at least one of the ISignals belonging to the ISignalGroup has a transferProperty defined all other ISignals belonging to the same ISignalGroup shall have a transferProperty defined as well. All of the transferProperties of the GroupSignals are considered.

Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
updateIndicationBitPosition	Integer	0..1	attr	The UpdateIndicationBit indicates to the receivers that the signal (or the signal group) was updated by the sender. Length is always one bit. The UpdateIndicationBitPosition attribute describes the position of the update bit within the SignalIPdu. For Signals of a ISignalGroup this attribute is irrelevant and shall be ignored. Note that the exact bit position of the updateIndicationBitPosition is linked to the value of the attribute packingByteOrder because the method of finding the bit position is different for the values mostSignificantByteFirst and mostSignificantByteLast. This means that if the value of packingByteOrder is changed while the value of updateIndicationBitPosition remains unchanged the exact bit position of updateIndicationBitPosition within the enclosing ISignalIPdu still undergoes a change.

Table 5.12: ISignalToIPduMapping

Enumeration	**TransferPropertyEnum**
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::FibexCore::CoreCommunication
Note	Transfer Properties of a Signal.
Literal	***Description***
pending	If the signal has the TransferProperty pending, then the function Com_SendSignal shall not perform a transmission of the IPdu associated with the signal.
triggered	The signal in the assigned IPdu is updated and a request for the IPdu's transmission is made.
triggeredOnChange	The signal in the assigned IPdu is updated and a request for the IPdus transmission is made only if the signal value is different from the already stored signal value.
triggeredOnChangeWithoutRepetition	The signal in the assigned IPdu is updated and a request for the IPdus transmission is made only if the signal value is different from the already stored signal value. In the DIRECT/N-TIMES or MIXED transmission mode (EventControlledTiming) the IPdu will be transmitted just once without a repetition, independent of the defined NumberOfRepeats.
triggeredWithoutRepetition	The signal in the assigned IPdu is updated and a request for the IPdu's transmission is made. In the DIRECT/N-TIMES or MIXED transmission mode (EventControlledTiming) the IPdu will be transmitted just once without a repetition, independent of the defined NumberOfRepeats.

Table 5.13: TransferPropertyEnum

[constr_3024] Usage of `triggeredWithoutRepetition` and `triggeredOnChangeWithoutRepetition` is not allowed for signal groups and group signals. ⌈ The values `triggeredWithoutRepetition` and `triggeredOnChange-`

WithoutRepetition shall not be used if the ISignalToIPduMapping refers to an ISignalGroup or an ISignal which is part of an ISignalGroup (group signal). ⌋

Class	**ISignalTriggering**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::FibexCore::CoreCommunication			
Note	A ISignalTriggering allows an assignment of ISignals to physical channels.			
Base	ARObject,Identifiable,MultilanguageReferrable,Referrable			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
iSignal	ISignal	0..1	ref	This reference shall be used if an ISignal is transported on the PhysicalChannel. This reference forms an XOR relationship with the ISignalTriggering-ISignalGroup reference. ISignalTriggerings for Group Signals (ISignals contained in the ISignalGroup) shall not be defined.
iSignalGroup	ISignalGroup	0..1	ref	This reference shall be used if an ISignalGroup is transported on the PhysicalChannel. This reference forms an XOR relationship with the ISignalTriggering-ISignal reference.
iSignalPort	ISignalPort	*	ref	References to the ISignalPort on every ECU of the system which sends and/or receives the ISignal. References for both the sender and the receiver side shall be included when the system is completely defined.

Table 5.14: ISignalTriggering

5.3.1 Big Endian and Little Endian memory layout of Pdus and Frames

The AUTOSAR system description provide means to specify how the memory layout looks like when signals are packed into Pdus and Pdus are packed into Frames. The layout of Pdus and Frames on different communication systems is out of scope of AUTOSAR. The specification of attributes Bit counting (monotone or sawtooth) and Bit order (decreasing or increasing)[2] is not supported by AUTOSAR. In AUTOSAR these attributes are fixed. The Bit counting is always "sawtooth" and the bit order is always "Decreasing".

When a signal is mapped into a Pdu only the packingByteOrder affects the memory layout of the signal inside the Pdu beginning with it´s start bit position.

Little endian stores the least significant byte first and begins with the least significant bit, i.e. loworder bit in the sequence (the least significant bit serves as start bit).

Big endian stores the most significant byte first and begins with the most significant bit, i.e. the bit with the greatest numerical value (the most significant bit serves as start bit).

[2]More details about Bit counting and Bit order can be found in ASAM FIBEX [7].

— AUTOSAR CONFIDENTIAL —

In both cases the bit positions in the mapped signals increase with the bit positions in the `ISignalIPdu` such that the bit 2^0 is mapped to position n in the `ISignalIPdu` and bit 2^1 is mapped to position n+1 and so on.

Example 5.6 shows the memory layout for Little Endian and Big Endian if an `ISignal` with a length of 10 bits is mappped into a Pdu. The start bit position is 5.

Little Endian byte order:

Byte	0								1							
Bit	7	6	5	4	3	2	1	0	15	14	13	12	11	10	9	8
Signal	2^2	2^1	2^0	-	-	-	-	-	-	2^9	2^8	2^7	2^6	2^5	2^4	2^3

Big Endian byte order:

Byte	0								1							
Bit	7	6	5	4	3	2	1	0	15	14	13	12	11	10	9	8
Signal	-	-	2^9	2^8	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0	-	-	-	-

Figure 5.6: PackingByteOrder Example

The mapping of Pdus into Frames is handled in the same way as the mapping of signals into Pdus.

5.4 PDUs

The PDU Router deploys ISignalIPdus, DcmIPdus, UserDefinedIPdus and MultiplexedIPdus onto different communication protocols. The PDU Router also determines if a transport protocol has to be used or not. Additional to the already mentioned Pdu Types the following types exist: NmPdu, NPdu, XcpPdu, UserDefinedPdus. These Pdus are not routed by the PDU Router. [3]

UserDefinedPdus and UserDefinedIPdus are used to describe PDU-based communication over Complex Communication Drivers. Chapter 5.18 provides a more detailed description of CDDs.

[3] There is one special gateway use case where a NPdu is routed by the Pdu Router. More details can be found in chapter 5.15.

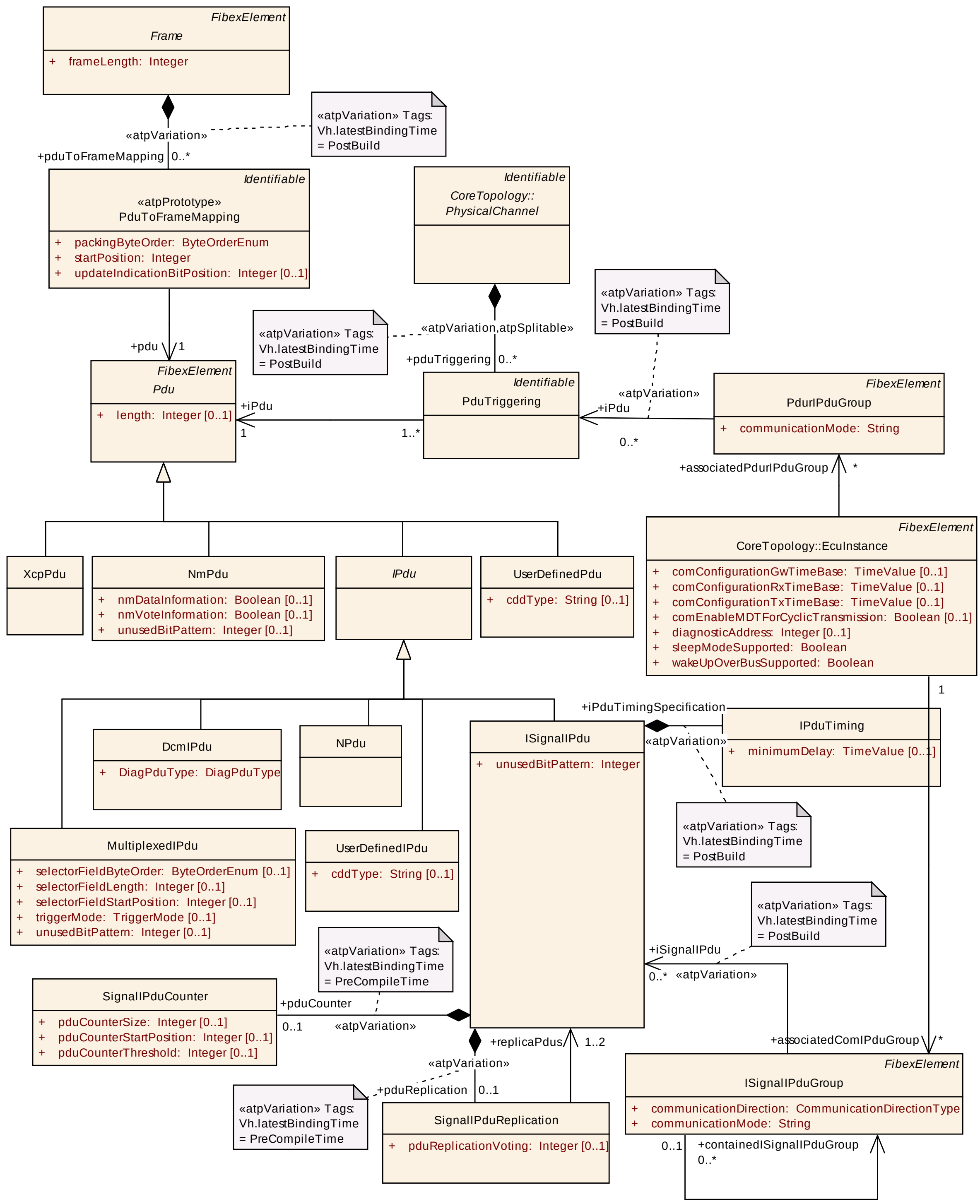

Figure 5.7: Pdus and the mapping into Frames (FibexCore: PDUOverview)

The `PDUToFrameMapping` element describes the mapping of `Pdus` to Frames and defines the position of a `Pdu` within a `Frame`. The distinction between the `Pdu` and `PduToFrameMapping` permits the usage of the same `Pdu` in different Frames.

A timing description `IPduTiming` can be aggregated directly by the `ISignalIPdu`. This timing description can be used for the Configuration of COM Transmission Modes. The `PduTriggering` describes on which channel the Pdu is transmitted. Timing requirements can be specified with the Timing Extension model. More details are described in chapter 1.4.4. Such Pdu timing requirements needs to be fulfilled by the timing specification on the Frame.

Class	**Pdu (abstract)**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::FibexCore::CoreCommunication			
Note	Collection of all Pdus that can be routed through a bus interface.			
Base	ARObject,CollectableElement,FibexElement,Identifiable,Multilanguage Referrable,PackageableElement,Referrable			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
length	Integer	0..1	attr	Pdu length in bytes. In case of dynamic length IPdus (containing a dynamical length signal), this value indicates the maximum data length. It should be noted that in former AUTOSAR releases (Rel 2.1, Rel 3.0, Rel 3.1, Rel 4.0 Rev. 1) this parameter was defined in bits. The Pdu length of zero bytes is allowed.

Table 5.15: Pdu

Class	**IPdu (abstract)**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::FibexCore::CoreCommunication			
Note	The IPdu (Interaction Layer Protocol Data Unit) element is used to sum up the IPdus of AUTOSAR COM, DCM and IPduM. These Pdus are routed by the PduR. In the AUTOSAR Layered Architecture the NPdu is not a specialization of an IPdu. The NPdu is located under the IPdu to support the low-level routing of NPdus. More details can be found in the NPdu class description.			
Base	ARObject,CollectableElement,FibexElement,Identifiable,Multilanguage Referrable,PackageableElement,Pdu,Referrable			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
–	–	–	–	–

Table 5.16: IPdu

Class	**ISignalIPdu**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::FibexCore::CoreCommunication			
Note	Represents the IPdus handled by Com. The ISignalIPdu assembled and disassembled in AUTOSAR COM consists of one or more signals. In case no multiplexing is performed this IPdu is routed to/from the Interface Layer. A maximum of one dynamic length signal per IPdu is allowed. **Tags:** atp.recommendedPackage=Pdus			
Base	ARObject,CollectableElement,FibexElement,IPdu,Identifiable,Multilanguage Referrable,PackageableElement,Pdu,Referrable			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***

Attribute	Datatype	Mul.	Kind	Note
iPduTiming Specificati on	IPduTiming	0..1	aggr	Timing specification for Com IPdus (Transmission Modes). atpVariation: The timing of a Pdu can vary. **Stereotypes:** atpVariation **Tags:** Vh.latestBindingTime=PostBuild
iSignalToP duMapping	ISignalToIPduM apping	*	aggr	Definition of SignalToIPduMappings included in the SignalIPdu. atpVariation: The content of a PDU can be variable. **Stereotypes:** atpVariation **Tags:** Vh.latestBindingTime=PostBuild
pduCounte r	SignalIPduCoun ter	0..1	aggr	An included Pdu counter is used to ensure that a sequence of Pdus is maintained. **Stereotypes:** atpVariation **Tags:** Vh.latestBindingTime=PreCompileTime
pduReplica tion	SignalIPduRepli cation	0..1	aggr	Pdu Replication is a form of redundancy where the data content of one ISignalIPdu (source) is transmitted inside a set of replica ISignalIPdus. These ISignalIPdus (copies) have different Pdu IDs, identical PduCounters, identical data content and are transmitted with the same frequency. **Stereotypes:** atpVariation **Tags:** Vh.latestBindingTime=PreCompileTime
unusedBit Pattern	Integer	1	attr	AUTOSAR COM and AUTOSAR IPDUM are filling not used areas of an IPDU with this bit-pattern. This attribute is mandatory to avoid undefined behavior. This byte-pattern will be repeated throughout the IPdu.

Table 5.17: ISignalIPdu

Class	**SignalIPduCounter**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::FibexCore::CoreCommunication			
Note	A PduCounter is included in a predefined set of Pdus and used to ensure that a sequence of Pdus is maintained. The counter is incremented when a Pdu is transmitted. The receivers check if the received Pdu is the next one in sequence.			
Base	ARObject			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
pduCounte rSize	Integer	0..1	attr	Size of PduCounter expressed in bits. Range: 1..8
pduCounte rStartPositi on	Integer	0..1	attr	Position of PduCounter expressed in bits. Note that PduCounter is not allowed to cross a byte border.
pduCounte rThreshold	Integer	0..1	attr	Threshold value of IPduCounter algorithm. See AUTOSAR COM Spec for more details.

Attribute	Datatype	Mul.	Kind	Note

Table 5.18: SignalIPduCounter

Class	SignalIPduReplication			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::FibexCore::CoreCommunication			
Note	PduReplication is a form of redundancy where the data content of one ISignalIPdu (source) is transmitted inside a set of replica ISignalIPdus. These ISignalIPdus (copies) have different Pdu IDs, identical PduCounters, identical data content and are transmitted with the same frequency.			
Base	ARObject			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**
pduReplicationVoting	Integer	0..1	attr	Number of identical IPdus needed for successful voting (1-3).
replicaPdus	ISignalIPdu	1..2	ref	Reference to replica Pdus of this IPdu.

Table 5.19: SignalIPduReplication

Class	NmPdu			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::FibexCore::CoreCommunication			
Note	Network Management Pdu **Tags:** atp.recommendedPackage=Pdus			
Base	ARObject,CollectableElement,FibexElement,Identifiable,MultilanguageReferrable,PackageableElement,Pdu,Referrable			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**
iSignalToIPduMapping	ISignalToIPduMapping	*	aggr	This optional aggregation is used to describe NmUserData that is transmitted in the NmPdu.
nmDataInformation	Boolean	0..1	attr	Defines if the Pdu contains NM Data.
nmVoteInformation	Boolean	0..1	attr	Defines if the Pdu contains NM Vote information.
unusedBitPattern	Integer	0..1	attr	AUTOSAR COM is filling not used areas of an Pdu with this bit-pattern. This attribute can only be used if the nmDataInformation attribute is set to true.

Table 5.20: NmPdu

Class	**NPdu**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::FibexCore::CoreCommunication			
Note	This is a Pdu of the Transport Layer. The main purpose of the TP Layer is to segment and reassemble IPdus. In case of a Pdu Gateway when the source and the target network are of the same kind (e.g. Can-to-Can routing) it is possible to optimize the routing. The incoming NPdu can be directly forwarded to the PduR and then be sent on the outbound bus without any (resource consuming) TP module involvement. To support this use case the NPdu is located under the IPdu. But in the AUTOSAR Layered Architecture the NPdu is not a specialization of an IPdu. **Tags:** atp.recommendedPackage=Pdus			
Base	ARObject,CollectableElement,FibexElement,IPdu,Identifiable,Multilanguage Referrable,PackageableElement,Pdu,Referrable			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**
–	–	–	–	–

Table 5.21: NPdu

Class	**XcpPdu**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::FibexCore::CoreCommunication			
Note	AUTOSAR XCP Pdu **Tags:** atp.recommendedPackage=Pdus			
Base	ARObject,CollectableElement,FibexElement,Identifiable,Multilanguage Referrable,PackageableElement,Pdu,Referrable			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**
–	–	–	–	–

Table 5.22: XcpPdu

Class	**DcmIPdu**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::FibexCore::CoreCommunication			
Note	Represents the IPdus handled by Dcm. **Tags:** atp.recommendedPackage=Pdus			
Base	ARObject,CollectableElement,FibexElement,IPdu,Identifiable,Multilanguage Referrable,PackageableElement,Pdu,Referrable			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**
DiagPduType	DiagPduType	1	attr	Attribute is used to distinguish a request from a response.

Table 5.23: DcmIPdu

Class	UserDefinedPdu			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::FibexCore::CoreCommunication			
Note	UserDefinedPdu allows to describe PDU-based communication over Complex Communication Drivers. If a new BSW module is added above the BusIf (e.g. a new Nm module) then this Pdu element shall be used to describe the communication. **Tags:** atp.recommendedPackage=Pdus			
Base	ARObject,CollectableElement,FibexElement,Identifiable,Multilanguage Referrable,PackageableElement,Pdu,Referrable			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
cddType	String	0..1	attr	This attribute defines the CDD that transmits or receives the UserDefinedIPdu. If several CDDs are defined this attribute is used to distinguish between them.

Table 5.24: UserDefinedPdu

Class	UserDefinedIPdu			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::FibexCore::CoreCommunication			
Note	UserDefinedIPdu allows to describe PDU-based communication over Complex Communication Drivers. If a new BSW module is added above the PduR (e.g. a Diagnostic Service) then this IPdu element shall be used to describe the communication. **Tags:** atp.recommendedPackage=Pdus			
Base	ARObject,CollectableElement,FibexElement,IPdu,Identifiable,Multilanguage Referrable,PackageableElement,Pdu,Referrable			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
cddType	String	0..1	attr	This attribute defines the CDD that transmits or receives the UserDefinedPdu. If several CDDs are defined this attribute is used to distinguish between them.

Table 5.25: UserDefinedIPdu

Class	≪atpPrototype≫ PduToFrameMapping			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::FibexCore::CoreCommunication			
Note	A PduToFrameMapping defines the composition of Pdus in each frame.			
Base	ARObject,Identifiable,MultilanguageReferrable,Referrable			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
packingByteOrder	ByteOrderEnum	1	attr	This attribute defines the order of the bytes of the Pdu and the packing into the Frame. The byte ordering "Little Endian" (MostSignificantByteLast) and "Big Endian" (MostSignificantByteFirst) can be selected. A mix between Little Endian and Big Endian within a Frame is not allowed (all PduToFrameMappings within a Frame must have the same packingByteOrder).

Attribute	Datatype	Mul.	Kind	Note
pdu	Pdu	1	ref	Reference to a I-Pdu, N-Pdu or NmPdu that is transmitted in the Frame.
startPosition	Integer	1	attr	This parameter is necessary to describe the byteposition of a Pdu within a Frame. Note that the absolute position of the Pdu in the Frame is determined by the definition of the packingByteOrder attribute. If Big Endian is specified, the start position indicates the bit position of the most significant bit in the Frame. If Little Endian is specified, the start position indicates the bit position of the least significant bit in the Frame.
updateIndicationBitPosition	Integer	0..1	attr	Indication to the receivers that the corresponding Pdu was updated by the sender. This attribute describes the position of the update bit in the frame that aggregates this PDUToFrameMapping. Length is always one bit. Note that the exact bit position of the updateIndicationBitPosition is linked to the value of the attribute packingByteOrder because the method of finding the bit position is different for the values mostSignificantByteFirst and mostSignificantByteLast. This means that if the value of packingByteOrder is changed while the value of updateIndicationBitPosition remains unchanged the exact bit position of updateIndicationBitPosition within the enclosing Frame still undergoes a change.

Table 5.26: PduToFrameMapping

Class	IPduTiming			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::FibexCore::CoreCommunication			
Note	AUTOSAR COM provides the possibility to define two different TRANSMISSION MODES for each IPdu. The Transmission Mode of an IPdu that is valid at a specific point in time is selected using the values of the signals that are mapped to this IPdu. For each IPdu a Transmission Mode Selector is defined. The Transmission Mode Selector is calculated by evaluating the conditions for a subset of signals (class TransmissionModeCondition in the System Template). The Transmission Mode Selector is defined to be true, if at least one Condition evaluates to true and is defined to be false, if all Conditions evaluate to false.			
Base	ARObject			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**
minimumDelay	TimeValue	0..1	attr	Minimum Delay in seconds between successive transmissions of this I-PDU, independent of the Transmission Mode.

Attribute	Datatype	Mul.	Kind	Note
transmissionModeDeclaration	TransmissionModeDeclaration	0..1	aggr	AUTOSAR COM allows configuring statically two different transmission modes for each I-PDU (True and False). The Transmission Mode Selector evaluates the conditions for a subset of signals and decides the transmission mode. It is possible to switch between the transmission modes during runtime.

Table 5.27: IPduTiming

Class	**PduTriggering**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::FibexCore::CoreCommunication			
Note	The PduTriggering describes on which channel the IPdu is transmitted. The Pdu routing by the PduR is only allowed for "IPdus" and not for NmPdus and XcpPdus. Depending on its relation to entities such channels and clusters it can be unambiguously deduced whether a fan-out is handled by the Pdu router or the Bus Interface. If the fan-out is specified between different clusters it shall be handled by the Pdu Router. If the fan-out is specified between different channels of the same cluster it shall be handled by the Bus Interface.			
Base	ARObject,Identifiable,MultilanguageReferrable,Referrable			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**
iPdu	Pdu	1	ref	Reference to the Pdu for which the PduTriggering is defined. One I-Pdu can be triggered on different channels (PduR fan-out). The Pdu routing by the PduR is only allowed for "IPdus" and not for NmPdus and XcpPdus. Nevertheless is the reference to the Pdu element necessary since the PduTriggering element is also used to specify the sending and receiving connections to EcuPorts.
iPduPort	IPduPort	*	ref	References to the IPduPort on every ECU of the system which sends and/or receives the I-PDU. References for both the sender and the receiver side shall be included when the system is completely defined.
iSignalTriggering	ISignalTriggering	*	ref	This reference provides the relationship to the ISignalTriggerings that are implemented by the PduTriggering. The reference is optional since no ISignalTriggering can be defined for DCM and Multiplexed Pdus. **Stereotypes:** atpVariation **Tags:** Vh.latestBindingTime=PostBuild

Table 5.28: PduTriggering

AUTOSAR COM provides a mechanism of starting/stopping COM PDU groups (`ISignalIPduGroup`).

Class	ISignalIPduGroup			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::FibexCore::CoreCommunication			
Note	The AUTOSAR COM Layer is able to start and to stop sending and receiving configurable groups of I-Pdus during runtime. An ISignalIPduGroup contains either ISignalIPdus or ISignalIPduGroups. When an ISignalIPduGroup containing one or more other ISignalIPduGroups is started the containedISignalIPduGroups shall also be started. When an ISignalIPduGroup containing one or more other ISignalIPduGroups is stopped the contained ISignalIPduGroups shall also be stopped. Only a two level hierarchy of ISignalIPdu groups is allowed. An ISignalIPdu group that is part of an ISignalIPdu group must not contain ISignalIPduGroups. **Tags:** atp.recommendedPackage=ISignaliPduGroup			
Base	ARObject,CollectableElement,FibexElement,Identifiable,Multilanguage Referrable,PackageableElement,Referrable			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**
communicationDirection	CommunicationDirectionType	1	attr	This attribute determines in which direction IPdus that are contained in this IPduGroup will be transmitted (communication direction can be either In or Out).
communicationMode	String	1	attr	This attribute defines the use-case for this ISignalIPduGroup (e.g. diagnostic, debugging etc.). For example, in a diagnostic mode all IPdus - which are not involved in diagnostic - are disabled. The use cases are not limited to a fixed enumeration and can be specified as a string.
containedISignalIPduGroup	ISignalIPduGroup	*	ref	An I-Pdu group can be included in other I-Pdu groups. Contained I-Pdu groups shall not be referenced by the EcuInstance.
iSignalIPdu	ISignalIPdu	*	ref	Reference to a set of Signal I-Pdus, which are contained in the ISignal I-Pdu Group. atpVariation: The content of a ISignal I-Pdu group can vary (->vehicle modes). **Stereotypes:** atpVariation **Tags:** Vh.latestBindingTime=PostBuild

Table 5.29: ISignalIPduGroup

Enumeration	**CommunicationDirectionType**
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::FibexCore::Core Communication
Note	Describes the communication direction.
Literal	*Description*
in	Reception (Input)
out	Transmission (Output)

Table 5.30: CommunicationDirectionType

[constr_3020] CommunicationDirection of containedIPduGroups ⌈ The value of the attribute `communicationDirection` of `containedIPduGroup` must be identical to the value of the attribute `communicationDirection` of the enclosing `IPduGroup`. ⌋

The AUTOSAR Pdu Router provides a mechanism of enabling/disabling of routing path groups (`PdurIPduGroup`).

Class	**PdurIPduGroup**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::FibexCore::CoreCommunication			
Note	The AUTOSAR PduR will enable and disable the sending of configurable groups of IPdus during runtime according to the AUTOSAR PduR specification. **Tags:** atp.recommendedPackage=PdurIPduGroups			
Base	ARObject,CollectableElement,FibexElement,Identifiable,Multilanguage Referrable,PackageableElement,Referrable			
Attribute	*Datatype*	*Mul.*	*Kind*	*Note*
communicationMode	String	1	attr	This attribute defines the use-case for this PduRIPduGroup. For example, in a diagnostic mode all IPdus - which are not involved in diagnostic - are disabled. The use cases are not limited to a fixed enumeration and can be specified as a string.
iPdu	PduTriggering	*	ref	Reference to a set of IPdus, which are contained in the PduR I-Pdu Group. If an IPdu is routed by the PduR to different destinations (PduR fan-out) than an PduTriggering for each destination is created in the System Template. To enable/disable a specific destination the PdurIPduGroup refers to the PduTriggering. atpVariation: The content of a PduR I-Pdu group can vary (->vehicle modes). **Stereotypes:** atpVariation **Tags:** Vh.latestBindingTime=PostBuild

Table 5.31: PdurIPduGroup

5.4.1 EndToEndProtection for ISignalIPduGroups

It is possible to protect the inter-ECU data exchange of safety-related ISignalGroups which are mapped into ISignalIPdus using protection mechanisms provided by E2E Library. It is possible to protect several ISignalGroups in one ISignalIPdu using several EndToEndProtectionISignalIPdu elements.

The E2EProtectionISignalIPdu element refers to the ISignalGroup that is to be protected and to the ISignalIPdu that transmits the protected ISignalGroup. The dataOffset in the E2EProtectionISignalIPdu element defines the starting position of the Array representation of the ISignalGroup.

The information how the referenced ISignalGroup shall be protected (through which E2E Profile and with which E2E settings) is defined in the EndToEndDescription element. All offset attributes of EndToEndDescription are relative to the dataOffset with respect to the ISignalIPdu (absolute position of the CRC = dataOffset + crcOffset). For more details, see End to End Library [17].

If the E2E Protection is done via COM Callouts then the EndToEndProtectionISignalIPdu shall be defined. If the E2E Protection is done in the E2E Wrapper then both EndToEndProtectionISignalIPdu and EndToEndProtectionVariablePrototype shall be defined. For more details, see Software Component Template specification [4].

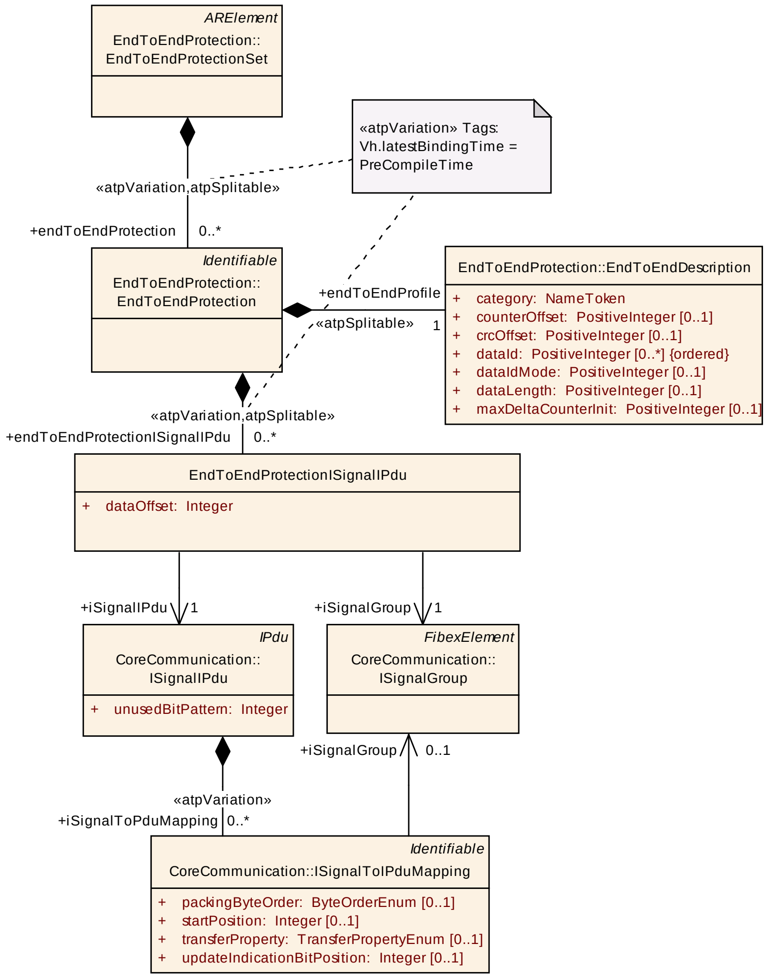

Figure 5.8: EndToEndProtection for COM IPdus

Class	**EndToEndProtectionSet**			
Package	M2::AUTOSARTemplates::SWComponentTemplate::EndToEndProtection			
Note	This represents a container for collection EndToEndProtectionInformation. **Tags:** atp.recommendedPackage=EndToEndProtectionSets			
Base	ARElement,ARObject,CollectableElement,Identifiable,Multilanguage Referrable,PackageableElement,Referrable			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**
endToEnd Protection	EndToEndProte ction	*	aggr	This is one particular EndToEndProtection. **Stereotypes:** atpSplitable; atpVariation **Tags:** Vh.latestBindingTime=PreCompileTime atp.Splitkey=shortName, variationPoint.shortLabel

Table 5.32: EndToEndProtectionSet

Class	EndToEndProtection			
Package	M2::AUTOSARTemplates::SWComponentTemplate::EndToEndProtection			
Note	This meta-class represents the ability to describe a particular end to end protection.			
Base	ARObject,Identifiable,MultilanguageReferrable,Referrable			
Attribute	Datatype	Mul.	Kind	Note
endToEndProfile	EndToEndDescription	1	aggr	This represents the particular EndToEndDescription. **Stereotypes:** atpSplitable **Tags:** atp.Splitkey=description
endToEndProtectionISignalIPdu	EndToEndProtectionISignalIPdu	*	aggr	Defines to which ISignalIPdu - ISignalGroup pair this EndToEndProtection shall apply. In case several ISignalGroups are used to transport the data (e.g. fan-out in the RTE) there may exist several EndToEndProtectionISignalIPdu definitions. **Stereotypes:** atpSplitable; atpVariation **Tags:** Vh.latestBindingTime=PreCompileTime atp.Splitkey=variationPoint.shortLabel
endToEndProtectionVariablePrototype	EndToEndProtectionVariablePrototype	*	aggr	Defines to which VariableDataPrototypes in the roles of one sender and one or more receivers this EndToEndprotection applies. It shall be possible to aggregate several EndToEndProtectionVariablePrototype in case additional hierarchical decompositions are introduced subsequently. In this case one particular PortPrototype is split into multiple PortPrototypes and connectors, all representing the same data entity. **Stereotypes:** atpSplitable; atpVariation **Tags:** Vh.latestBindingTime=PreCompileTime atp.Splitkey=shortLabel, variationPoint.shortLabel

Table 5.33: EndToEndProtection

Class	EndToEndProtectionISignalIPdu			
Package	M2::AUTOSARTemplates::SystemTemplate::EndToEndProtection			
Note	It is possible to protect the inter-ECU data exchange of safety-related ISignalGroups at the level of COM IPdus using protection mechanisms provided by E2E Library. For each ISignalGroup to be protected, a separate EndToEndProtectionISignalIPdu element must be created within the EndToEndProtectionSet. The EndToEndProtectionISignalIPdu element refers to the ISignalGroup that is to be protected and to the ISignalIPdu that transmits the protected ISignalGroup. The information how the referenced ISignalGroup shall be protected (through which E2E Profile and with which E2E settings) is defined in the EndToEndDescription element.			
Base	ARObject			
Attribute	Datatype	Mul.	Kind	Note

Attribute	Datatype	Mul.	Kind	Note
dataOffset	Integer	1	attr	This attribute defines the beginning offset (in bits) of the Array representation of the Signal Group (including CRC, counter and application signal group) in the IPdu. This attribute is mandatory and the dataOffset shall always be defined.
iSignalGroup	ISignalGroup	1	ref	Reference to the ISignalGroup that is to be protected.
iSignalIPdu	ISignalIPdu	1	ref	Reference to the ISignalIPdu that transmits the protected ISignalGroup.

Table 5.34: EndToEndProtectionISignalIPdu

Class	EndToEndDescription			
Package	M2::AUTOSARTemplates::SWComponentTemplate::EndToEndProtection			
Note	This meta-class contains information about end-to-end protection. The set of applicable attributes depends on the actual value of the category attribute of EndToEndProtection.			
Base	ARObject			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**
category	NameToken	1	attr	The category represents the identification of the concrete E2E profile. The applicable values are specified in a semantic constraint and determine the applicable attributes of EndToEndDescription. **Tags:** xml.sequenceOffset=-100
counterOffset	PositiveInteger	0..1	attr	Bit offset of Counter from the beginning of the Array representation of the Signal Group/VariableDataPrototype (MSB order, bit numbering: bit 0 is the least important). The offset shall be a multiplicity of 4 and it should be 8 whenever possible. For example, offset 8 means that the counter will take the low nibble of the byte 1, i.e. bits 8 .. 11. If counterOffset is not present the value is defined by the selected profile. **Tags:** xml.sequenceOffset=-50
crcOffset	PositiveInteger	0..1	attr	Bit offset of CRC from the beginning of the Array representation of the Signal Group/VariableDataPrototype (MSB order, bit numbering: bit 0 is the least important). The offset shall be a multiplicity of 8 and it should be 0 whenever possible. For example, offset 8 means that the CRC will take the byte 1, i.e. bits 8..15. If crcOffset is not present the value is defined by the selected profile. **Tags:** xml.sequenceOffset=-60

Attribute	Datatype	Mul.	Kind	Note
dataId (ordered)	PositiveInteger	*	attr	This represents a unique numerical identifier. Note: ID is used for protection against masquerading. The details concerning the maximum number of values (this information is specific for each E2E profile) applicable for this attribute are controlled by a semantic constraint that depends on the category of the EndToEndProtection. **Tags:** xml.sequenceOffset=-90
dataIdMode	PositiveInteger	0..1	attr	There are three inclusion modes how the implicit two-byte Data ID is included in the one-byte CRC: • dataIDMode = 0: Two bytes are included in the CRC (double ID configuration) This is used in variant 1A. • dataIDMode = 1: One of the two bytes byte is included, alternating high and low byte, depending on parity of the counter (alternating ID configuration). For even counter low byte is included; For odd counters the high byte is included. This is used in variant 1B. • dataIDMode = 2: Only low byte is included, high byte is never used. This is applicable if the IDs in a particular system are 8 bits. **Tags:** xml.sequenceOffset=-85
dataLength	PositiveInteger	0..1	attr	This attribute represents the length of the Array representation of the Signal Group/VariableDataPrototype including CRC and Counter in bits. **Tags:** xml.sequenceOffset=-80
maxDeltaCounterInit	PositiveInteger	0..1	attr	Initial maximum allowed gap between two counter values of two consecutively received valid Data, i.e. how many subsequent lost data is accepted. For example, if the receiver gets Data with counter 1 and MaxDeltaCounterInit is 1, then at the next reception the receiver can accept Counters with values 2 and 3, but not 4. Note that if the receiver does not receive new Data at a consecutive read, then the receiver increments the tolerance by 1. **Tags:** xml.sequenceOffset=-70

Table 5.35: EndToEndDescription

— AUTOSAR CONFIDENTIAL —

The maxDeltaCounterInit value can also be specified in the ReceiverComSpec. This allows the definition of a receiver specific value. The value of the attribute maxDeltaCounterInit provided on the ReceiverComSpec overrides a possible value in the EndToEndDescription class. More details can be found in the Software Component Template specification [4].

The supported E2E profiles (possible values of category in EndToEndDescription) are described in the Software Component Template [4] and the End to End Library [17].

5.5 Frames

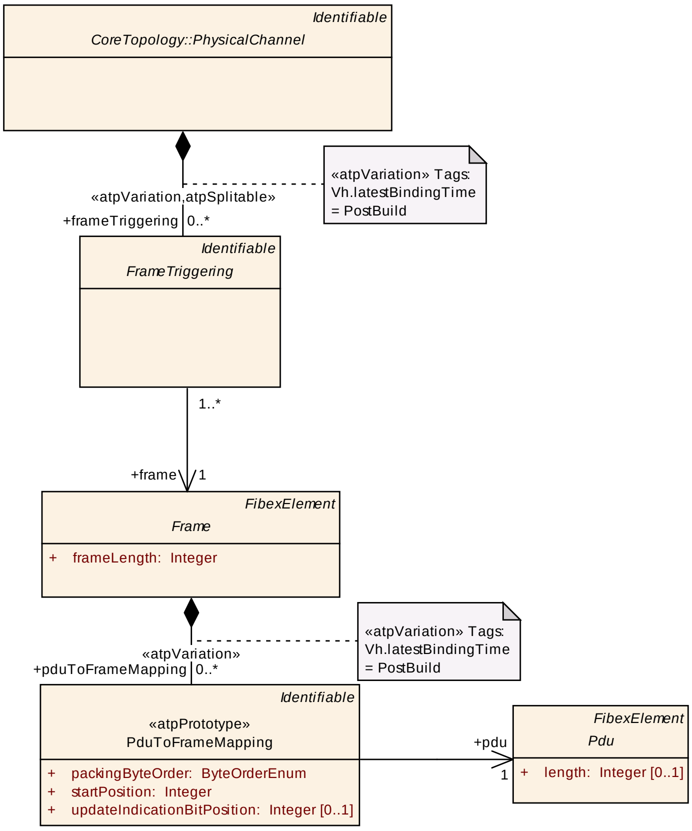

Figure 5.9: Frame Overview (FibexCore: FrameOverview)

`Frames` can be defined independently of communication clusters. On the communication channel the `Frame` is represented by the referencing `FrameTriggering`. The `FrameTriggering` defines a frame's send behavior and identification on a certain channel.

[constr_3012] Overlapping of Pdus is prohibited ⌈ `Pdus` mapped to a `Frame` MUST NOT overlap. ⌋

[constr_3013] Frame length shall not be exceeded ⌈ The combined length of all `Pdus` that are mapped into an `Frame` shall not exceed the defined `Frame` length. ⌋

[constr_3014] Overlapping of updateIndicationBits for Pdus is prohibited ⌈ The `updateIndicationBitPosition` for a `Pdu` in a `Frame` MUST NOT overlap with other `updateIndicationBitPositions` and `Pdu` Locations. ⌋

Class	**Frame (abstract)**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::FibexCore::CoreCommunication			
Note	Data frame which is sent over a communication medium. This element describes the pure Layout of a frame sent on a channel.			
Base	ARObject,CollectableElement,FibexElement,Identifiable,Multilanguage Referrable,PackageableElement,Referrable			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***

Attribute	Datatype	Mul.	Kind	Note
frameLength	Integer	1	attr	The used length (in bytes) of the referencing frame. Should not be confused with a static byte length reserved for each frame by some platforms (e.g. FlexRay). The frameLength of zero bytes is allowed.
pduToFrameMapping	PduToFrameMapping	*	aggr	A frames layout as a sequence of Pdus. atpVariation: The content of a frame can be variable. **Stereotypes:** atpVariation **Tags:** Vh.latestBindingTime=PostBuild

Table 5.36: Frame

Class	FrameTriggering (abstract)			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::FibexCore::CoreCommunication			
Note	The FrameTriggering describes the instance of a frame sent on a channel and defines the manner of triggering (timing information) and identification of a frame on the channel, on which it is sent. For the same frame, if FrameTriggerings exist on more than one channel of the same cluster the fan-out/in is handled by the Bus interface.			
Base	ARObject,Identifiable,MultilanguageReferrable,Referrable			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**
frame	Frame	1	ref	One frame can be triggered several times, e.g. on different channels. If a frame has no frame triggering, it won't be sent at all. A frame triggering has assigned exactly one frame, which it triggers.
framePort	FramePort	*	ref	References to the FramePort on every ECU of the system which sends and/or receives the frame. References for both the sender and the receiver side shall be included when the system is completely defined.
pduTriggering	PduTriggering	*	ref	This reference provides the relationship to the PduTriggerings that are implemented by the FrameTriggering. The reference is optional since no PduTriggering can be defined for NmPdus and XCP Pdus. **Stereotypes:** atpVariation **Tags:** Vh.latestBindingTime=PostBuild

Table 5.37: FrameTriggering

5.6 I-Pdu Multiplexer

Multiplexing is used to transport varying Com I-Pdus at the same position in a single multiplexed I-Pdu. A multiplexed I-Pdu consists a dynamic part, a selector field and an optional static part. According to the value of the selector field the dynamic part can have a different layout. For each alternative there is one COM I-Pdu that is transmitted in the dynamic part. The static part of the multiplexed I-Pdu is the same regardless of the selector field and consists of one Com I-Pdu.

The `MultiplexedIPdu` element contains attributes that describe the position and the length of a selector within an IPdu. A selector is a bitfield of certain length, by the value of which the corresponding data region of the dynamic part must be interpreted dynamically, i.e. at run-time.

[constr_3007] `SelectorFieldCodes` for dynamic part alternatives ⌈ The `selectorFieldCodes` for the dynamic part alternatives within one `MultiplexedIPdu` shall differ from each other. ⌋

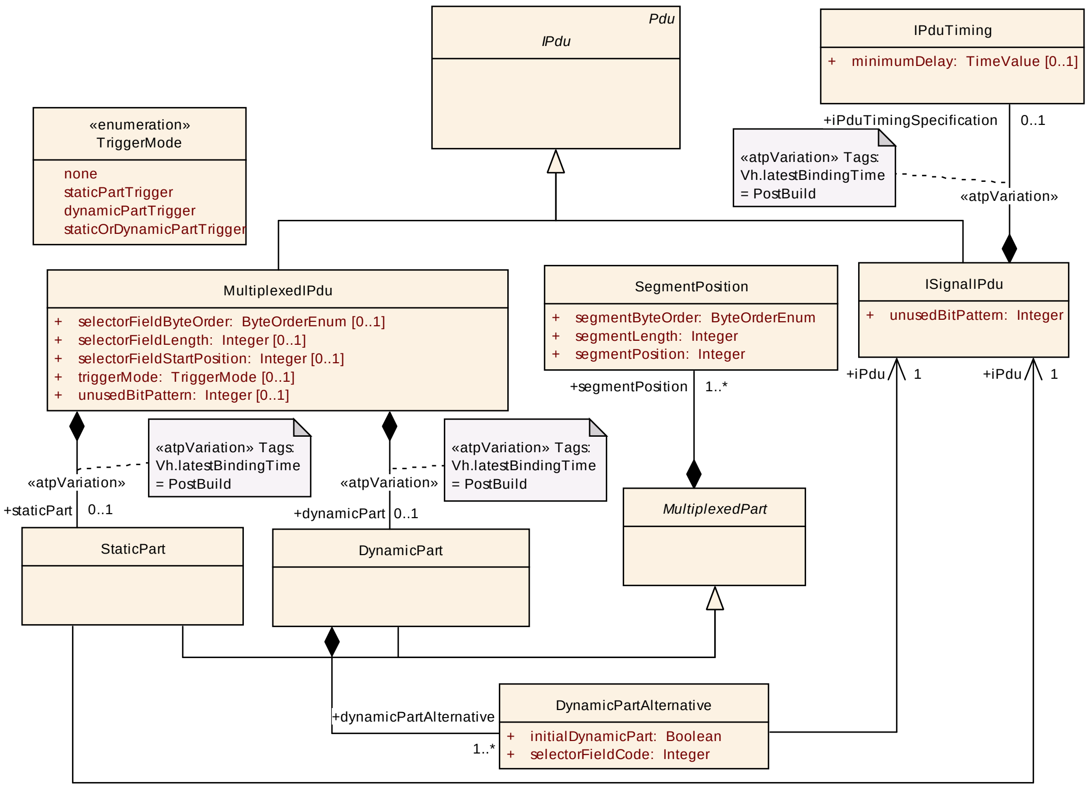

Figure 5.10: I-Pdu Multiplexer (FibexCore: IPDUMultiplexerOverview)

Enumeration	**TriggerMode**
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::FibexCore::Core Communication
Note	IPduM can be configured to send a transmission request for the new multiplexed I-PDU to the PDU-Router because of conditions/ modes.
Literal	***Description***
dynamicPart Trigger	IPduM sends a transmission request to the PduR if a dynamic part is received.
none	IPduM does not trigger transmission because of receiving anything of this IPdu in case of TriggerTransmit.
staticOrDy-namicPart Trigger	IPduM sends a transmission request to the PduR if a static or dynamic part is received.
staticPart Trigger	IPduM sends a transmission request to the PduR if a static part is received.

Table 5.38: TriggerMode

Class	**MultiplexedIPdu**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::FibexCore::CoreCommunication			
Note	A MultiplexedPdu (i.e. NOT a COM I-PDU) contains a DynamicPart, an optional StaticPart and a selectorField. In case of multiplexing this IPdu is routed between the Pdu Multiplexer and the Interface Layer. A multiplexer is used to define variable parts within an IPdu that may carry different signals. The receivers of such a IPdu can determine which signalPdus are transmitted by evaluating the selector field, which carries a unique selector code for each sub-part. **Tags:** atp.recommendedPackage=Pdus			
Base	ARObject,CollectableElement,FibexElement,IPdu,Identifiable,Multilanguage Referrable,PackageableElement,Pdu,Referrable			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
dynamicPa rt	DynamicPart	0..1	aggr	According to the value of the selector field some parts of the IPdu have a different layout. In a complete System Description a MultiplexedIPdu must contain a DynamicPart. If a MultiplexedPdu is received by a Pdu Gateway and is not delivered to the IPduM but routed directly to a bus interface then the content of the MulitplexedPdu doesn't need to be described in the System Extract/Ecu Extract. To support this use case the multiplicity is set to 0..1. atpVariation: Content of a multiplexed PDU can vary. **Stereotypes:** atpVariation **Tags:** Vh.latestBindingTime=PostBuild

Attribute	Datatype	Mul.	Kind	Note
selectorFieldByteOrder	ByteOrderEnum	0..1	attr	This attribute defines the order of the bytes of the selectorField and the packing into the MultiplexedIPdu. The byte ordering "Little Endian" (MostSignificantByteLast) and "Big Endian" (MostSignificantByteFirst) can be selected. A mix between Little Endian and Big Endian within a MultiplexedIPdu (staticPart, dynamicPart, selectorField) is not allowed. In a complete System Description this attribute is mandatory. If a MultiplexedPdu is received by a Pdu Gateway and is not delivered to the IPduM but routed directly to a bus interface then the content of the MulitplexedPdu doesn't need to be described in the System Extract/Ecu Extract. To support this use case the multiplicity is set to 0..1.
selectorFieldLength	Integer	0..1	attr	The size in bits of the selector field shall be configurable in a range of 0-2031 bits. In a complete System Description this attribute is mandatory. If a MultiplexedPdu is received by a Pdu Gateway and is not delivered to the IPduM but routed directly to a bus interface then the content of the MulitplexedPdu doesn't need to be described in the System Extract/Ecu Extract. To support this use case the multiplicity is set to 0..1.
selectorFieldStartPosition	Integer	0..1	attr	This parameter is necessary to describe the position of the selector field within the IPdu. Note that the absolute position of the selectorField in the MultiplexedIPdu is determined by the definition of the selectorFieldByteOrder attribute of the Multiplexed Pdu. If Big Endian is specified, the start position indicates the bit position of the most significant bit in the IPdu. If Little Endian is specified, the start position indicates the bit position of the least significant bit in the IPdu. In AUTOSAR the bit counting is always set to "sawtooth" and the bit order is set to "Decreasing". In a complete System Description this attribute is mandatory. If a MultiplexedPdu is received by a Pdu Gateway and is not delivered to the IPduM but routed directly to a bus interface then the content of the MulitplexedPdu doesn't need to be described in the System Extract/Ecu Extract. To support this use case the multiplicity is set to 0..1.

Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
staticPart	StaticPart	0..1	aggr	The static part of the multiplexed IPdu is the same regardless of the selector field. The static part is optional. atpVariation: Content of a multiplexed PDU can vary. **Stereotypes:** atpVariation **Tags:** Vh.latestBindingTime=PostBuild
triggerMode	TriggerMode	0..1	attr	IPduM can be configured to send a transmission request for the new multiplexed IPdu to the PDU-Router because of the trigger conditions/ modes that are described in the TriggerMode enumeration. In a complete System Description this attribute is mandatory. If a MultiplexedPdu is received by a Pdu Gateway and is not delivered to the IPduM but routed directly to a bus interface then the content of the MulitplexedPdu doesn't need to be described in the System Extract/Ecu Extract. To support this use case the multiplicity is set to 0..1.
unusedBitPattern	Integer	0..1	attr	AUTOSAR COM and AUTOSAR IPDUM are filling not used areas of an IPdu with this bit-pattern. This attribute is mandatory to avoid undefined behavior. This byte-pattern will be repeated throughout the IPdu. In a complete System Description this attribute is mandatory. If a MultiplexedPdu is received by a Pdu Gateway and is not delivered to the IPduM but routed directly to a bus interface then the content of the MulitplexedPdu doesn't need to be described in the System Extract/Ecu Extract. To support this use case the multiplicity is set to 0..1.

Table 5.39: MultiplexedIPdu

Class	**StaticPart**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::FibexCore::CoreCommunication			
Note	Some parts/signals of the I-PDU may be the same regardless of the selector field. Such a part is called static part. The static part is optional.			
Base	ARObject,MultiplexedPart			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
iPdu	ISignalIPdu	1	ref	Reference to a Com IPdu which is routed to the IPduM module and is combined to a multiplexedPdu.

Table 5.40: StaticPart

Class	DynamicPart			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::FibexCore::CoreCommunication			
Note	Dynamic part of a multiplexed I-Pdu. Reserved space which is used to transport varying SignalIPdus at the same position, controlled by the corresponding selectorFieldCode.			
Base	ARObject,MultiplexedPart			
Attribute	Datatype	Mul.	Kind	Note
dynamicPartAlternative	DynamicPartAlternative	1..*	aggr	Com IPdu alternatives that are transmitted in the Dynamic Part of the MultiplexedIPdu.

Table 5.41: DynamicPart

Class	DynamicPartAlternative			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::FibexCore::CoreCommunication			
Note	One of the Com IPdu alternatives that are transmitted in the Dynamic Part of the MultiplexedIPdu. The selectorFieldCode specifies which Com IPdu is contained in the DynamicPart within a certain transmission of a multiplexed PDU.			
Base	ARObject			
Attribute	Datatype	Mul.	Kind	Note
iPdu	ISignalIPdu	1	ref	Reference to a Com IPdu which is routed to the IPduM module and is combined to a multiplexedPdu.
initialDynamicPart	Boolean	1	attr	Dynamic part that shall be used to initialize this multiplexed IPdu. Constraint: Only one "DynamicPartAlternative" in a "DynamicPart" shall be the initialDynamicPart.
selectorFieldCode	Integer	1	attr	The selector field is part of a multiplexed IPdu. It consists of contiguous bits. The value of the selector field selects the layout of the multiplexed part of the IPdu.

Table 5.42: DynamicPartAlternative

Class	MultiplexedPart (abstract)			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::FibexCore::CoreCommunication			
Note	The StaticPart and the DynamicPart have common properties. Both can be separated in multiple segments within the multiplexed PDU.			
Base	ARObject			
Attribute	Datatype	Mul.	Kind	Note
segmentPosition	SegmentPosition	1..*	aggr	The StaticPart and the DynamicPart can be separated in multiple segments within the multiplexed PDU. Therefore the StaticPart and the DynamicPart can contain multiple SegmentPositions.

Table 5.43: MultiplexedPart

Class	**SegmentPosition**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::FibexCore::CoreCommunication			
Note	The StaticPart and the DynamicPart can be separated in multiple segments within the multiplexed PDU. The ISignalIPdus are copied bit by bit into the MultiplexedIPdu. If the space of the first segment is 5 bits large than the first 5 bits of the ISignalIPdu are copied into this first segment and so on.			
Base	ARObject			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**
segmentByteOrder	ByteOrderEnum	1	attr	This attribute defines the order of the bytes of the segment and the packing into the MultiplexedIPdu. The byte ordering "Little Endian" (MostSignificantByteLast) and "Big Endian" (MostSignificantByteFirst) can be selected. A mix between Little Endian and Big Endian within a MultiplexedIPdu (staticPart, dynamicPart, selectorField) is not allowed.
segmentLength	Integer	1	attr	Data Length of the segment in bits.
segmentPosition	Integer	1	attr	Segments bit position relatively to the beginning of a multiplexed IPdu. Note that the absolute position of the segment in the MultiplexedIPdu is determined by the definition of the segmentByteOrder attribute of the SegmentPosition. If Big Endian is specified, the start position indicates the bit position of the most significant bit in the IPdu. If Little Endian is specified, the start position indicates the bit position of the least significant bit in the IPdu. In AUTOSAR the bit counting is always set to "sawtooth" and the bit order is set to "Decreasing".

Table 5.44: SegmentPosition

Figure 5.11 shows an example of an IPdu Multiplexer. The static part of the multiplexed IPdu contains ComIPduA. The value of the selector field in the dynamic part decides which content is transmitted. ComIPduB is transmitted if the selector field value is "0". ComIPduC is transmitted if the selector field value is "1".

The static and the dynamic part can consist of more than one element. These sub parts of the static or dynamic parts are called segments. In Figure 5.11 the dynamic Part is segmented into two parts. More details can be found in [18].

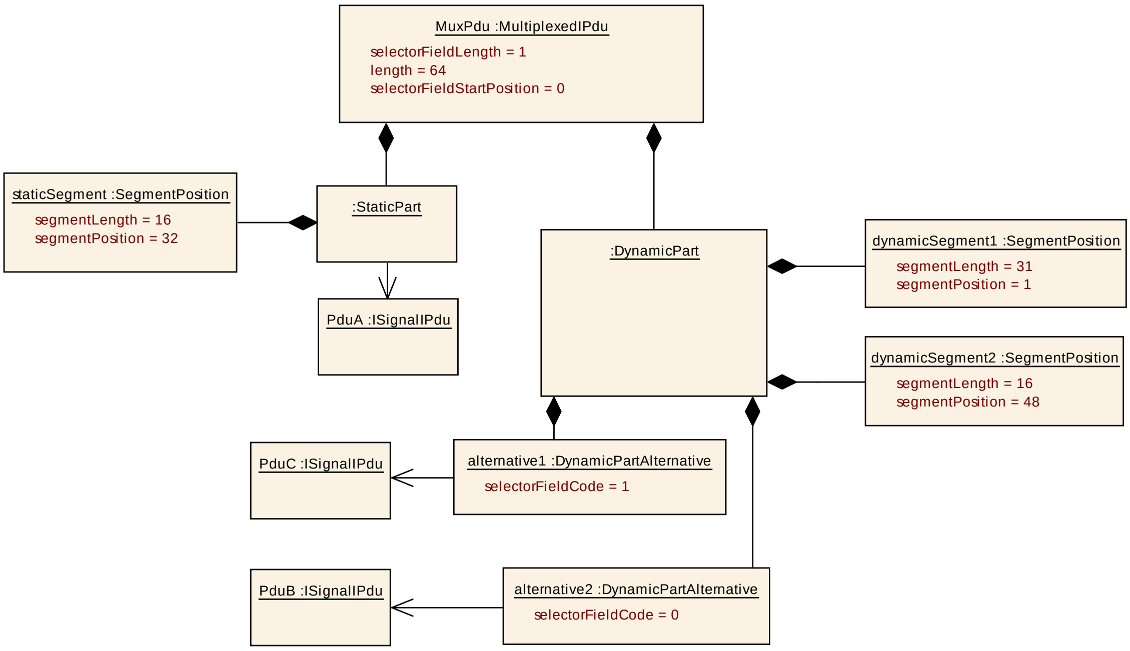

Figure 5.11: I-Pdu Multiplexer Example

[constr_3017] Length of multiplexed Pdu shall not be exceeded. ⌈The sum of included IPdus (static Part and dynamic Part) plus the length of the switch shall be smaller or equal than the length of the containing multiplexer Pdu. ⌋

5.6.1 I-Pdu Multiplexer in System Extract/ECU Extract

The processing in the ECU determine the description of `MultiplexedIPdu's` in the System Extract/Ecu Extract. In case that a Gateway ECU only routes a `MultiplexedIPdu` without being interested in the content leads to a reduced description in the System Extract/ECU Extract. The following items describe the different scenarios and the consequences for the System Extract/ECU Extract description. A complete System Description contains all informations (scenario 1).

1) Sending or receiving a `MultiplexedIPdu`

- all attributes of the `MultiplexedIPdu` are mandatory
- aggregated `dynamicPart` with associated `ISignalIPdus` is mandatory
- a `PduTriggering` shall be defined for the `MultiplexedIPdu`
- a `PduTriggering` shall be defined for all included `ISignalIPdus` in the `dynamicPart` and `staticPart`

The initial ECU Configuration Generator configures COM, PduR, IpduM and lower layers with the information from the System Extract/ECU Extract.

2) Only gatewaying a `MultiplexedIPdu`

- staticPart and dynamicPart definitions shall be omitted, thus no ISignalIPdu description shall be included
- all attributes of the MultiplexedIPdu shall be omitted.
- a PduTriggering shall be defined only for the gatewayed MultiplexedIPdu
- an IPduMapping between the source and the target PduTriggerings shall be defined

The initial ECU Configuration Generator configures PduR and lower layers with the information from the System Extract/ECU Extract.

3) Receiving and gatewaying a MultiplexedIPdu

- all attributes of the MultiplexedIPdu are mandatory
- aggregated dynamicPart with associated ISignalIPdus is mandatory
- a PduTriggering shall be defined for the MultiplexedIPdu
- an IPduMapping between the source and the target PduTriggerings shall be defined
- a PduTriggering shall be defined for all included ISignalIPdus in the dynamicPart and staticPart

The initial ECU Configuration Generator configures Com, PduR, IpduM and lower layers with the information from the System Extract/ECU Extract.

5.7 Frame Timing

Frame timing defines the time behavior of Frames. The description of the Timing must be precise enough that the System Generator can calculate the bus load and the resulting time for the transmission of a frame.

In the Basic Software the timing of bus frames can be controlled by send requests of the RTE in combination with the Transmission Mode and Transfer Property parameters in COM. On the other hand the timing can be controlled by the FlexRay Interface and LIN Interface.

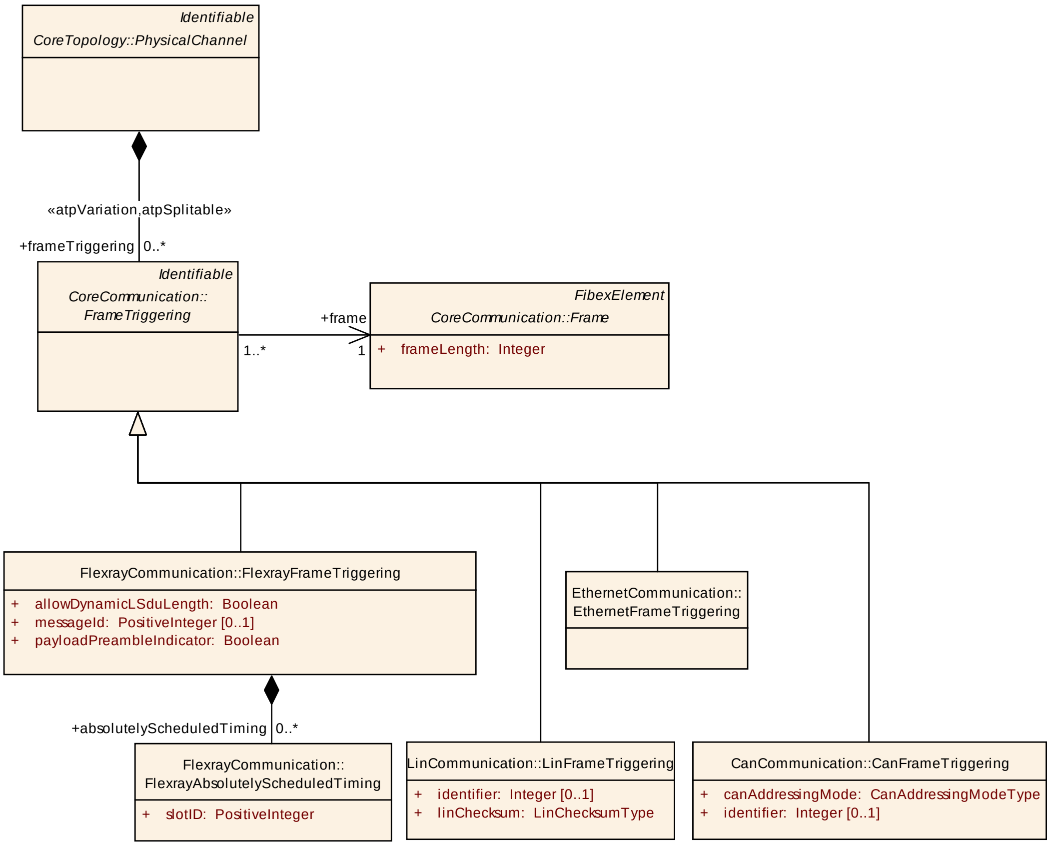

Figure 5.12: Frame Triggering

In FlexRay each frame is identified by its slot id and communication cycle. The `FlexrayAbsolutelyScheduledTiming` is described in chapter 5.8. Schedule tables organize the Timings of the frames for LIN (chapter 5.9).

5.8 FlexRay specific description

FlexRay is a time triggered communication protocol that provides a deterministic part (static segment) as well as a non-deterministic part (dynamic segment).

In the following, the elements will be specified, which are necessary to describe the FlexRay Frames and the FlexRay Communication.

FlexRay static channel parameters: Each frame in FlexRay is identified by its slot id and communication cycle. In the static segment all communication slots are of identical, statically configured duration and all frames are of identical, statically configured length.

The sending behavior where the exact time for the frames transmission is guaranteed is provided in the System Template/FIBEX by the usage of `FlexrayAbsolutelyScheduledTiming`.

In the cycle counter field of every frame, the current value of the cycle counter is transmitted (see FlexRay frame format). This value is incremented at the beginning of each new cycle, ranging from 0 to 63, and is reset to 0 after a sequence of 64 cycles. In the static segment frames can be sent multiple times within one communication cycle. For describing this case multiple `FlexrayAbsolutelyScheduledTiming` have to be used.

FlexRay dynamic channel parameters: In the dynamic segment the duration of communication slots may vary in order to accommodate frames of varying length. Furthermore, in the dynamic part, the slot id is equivalent to a priority. The higher the number the lower is the priority. But the frames in the static and in the dynamic channel have the same format. Each FlexRay Frame is identified by its slot id and communication cycle. A description is provided by the usage of `FlexrayAbsolutelyScheduledTiming`.

If the behavior of a FlexRay frame is cyclic or event triggered, a timing requirement can be specified with the Timing Extension model. More details are described in chapter 1.4.4. Such a timing requirement must be fulfilled by the timing specification on the frame.

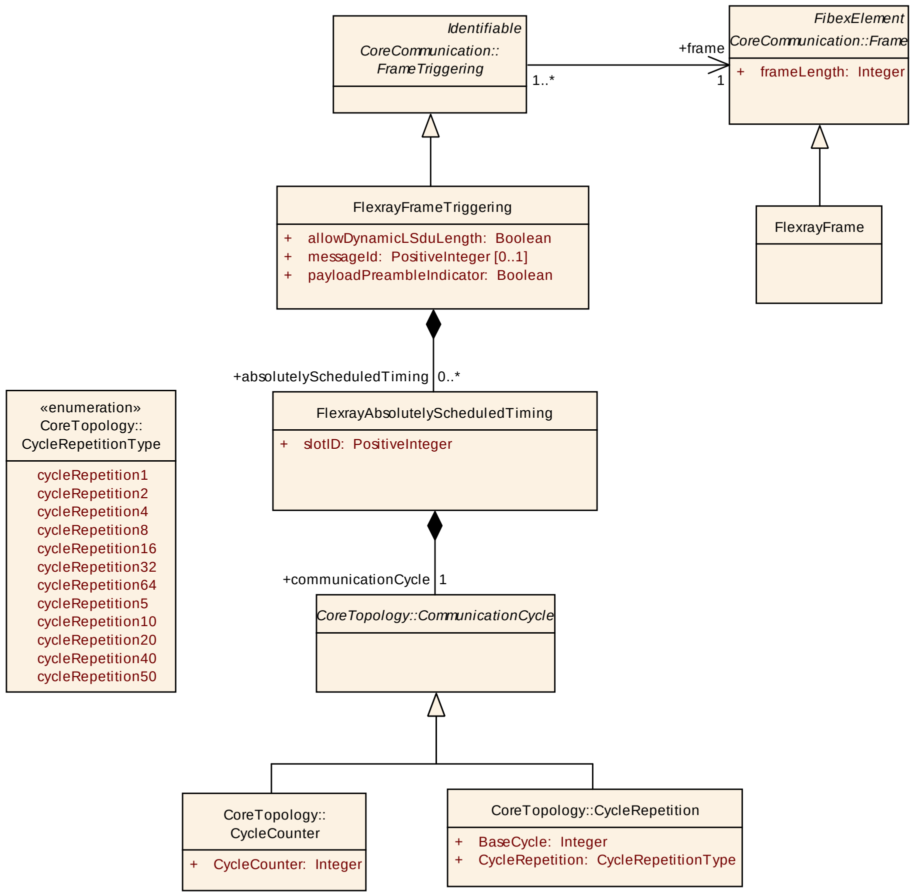

Figure 5.13: FlexRay Absolutely Scheduled Timing (Fibex4FlexRay:FlexrayAbsolutelyScheduledTiming)

Class	**FlexrayFrame**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Flexray::Flexray Communication			
Note	FlexRay specific Frame element. **Tags:** atp.recommendedPackage=Frames			
Base	ARObject,CollectableElement,FibexElement,Frame,Identifiable,Multilanguage Referrable,PackageableElement,Referrable			
Attribute	*Datatype*	*Mul.*	*Kind*	*Note*
–	–	–	–	–

Table 5.45: FlexrayFrame

Class	**FlexrayFrameTriggering**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Flexray::Flexray Communication			
Note	FlexRay specific attributes to the FrameTriggering			
Base	ARObject,FrameTriggering,Identifiable,MultilanguageReferrable,Referrable			
Attribute	*Datatype*	*Mul.*	*Kind*	*Note*

Attribute	Datatype	Mul.	Kind	Note
absolutely Scheduled Timing	FlexrayAbsolute lyScheduledTim ing	*	aggr	Specification of a sending behaviour where the exact time for the frames transmission is guaranteed.
allowDyna micLSduLe ngth	Boolean	1	attr	Allows L-PDU length reduction and indicates that the related CC buffer has to be reconfigured for the actual length and Header-CRC before transmission of the L-PDU. If this attribute is set to true than the referenced Frame length attribute defines the max. length.
messageId	PositiveInteger	0..1	attr	The first two bytes of the payload segment of the FlexRay frame format for frames transmitted in the dynamic segment can be used as receiver filterable data called the message ID.
payloadPr eambleIndi cator	Boolean	1	attr	Switching the Payload Preamble bit.

Table 5.46: FlexrayFrameTriggering

Class	**FlexrayAbsolutelyScheduledTiming**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Flexray::Flexray Communication			
Note	Each frame in FlexRay is identified by its slot id and communication cycle. A description is provided by the usage of AbsolutelyScheduledTiming. In the static segment a frame can be sent multiple times within one communication cycle. For describing this case multiple AbsolutelyScheduledTimings have to be used. The main use case would be that a frame is sent twice within one communication cycle.			
Base	ARObject			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
communic ationCycle	Communication Cycle	1	aggr	The communication cycle where the frame is sent.
slotID	PositiveInteger	1	attr	In the static part the SlotID defines the slot in which the frame is transmitted. The SlotID also determines, in combination with FlexrayCluster::numberOfStaticSlots, whether the frame is sent in static or dynamic segment. In the dynamic part, the slot id is equivalent to a priority. Lower dynamic slot ids are all sent until the end of the dynamic segment. Higher numbers, which were ignored that time, have to wait one cycle and then must try again. minValue: 1 maxValue: 2047

Table 5.47: FlexrayAbsolutelyScheduledTiming

Class	**CommunicationCycle (abstract)**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::FibexCore::CoreTopology			
Note	The communication cycle where the frame is sent.			
Base	ARObject			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
–	–	–	–	–

Table 5.48: CommunicationCycle

The communication cycle can be described by the CycleCounterType or by the CycleRepetitionType:

Class	**CycleCounter**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::FibexCore::CoreTopology			
Note	The communication cycle where the frame is send is described by the attribute "cycleCounter".			
Base	ARObject,CommunicationCycle			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
CycleCounter	Integer	1	attr	The communication cycle where the frame described by this timing is sent. If a timing is given in this way the referencing cluster must specify the NUMBER-OF-CYCLES as upper bound and point of total repetition. This value is incremented at the beginning of each new cycle, ranging from 0 to 63, and is reset to 0 after a sequence of 64 cycles.

Table 5.49: CycleCounter

Class	**CycleRepetition**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::FibexCore::CoreTopology			
Note	The communication cycle where the frame is send is described by the attributes baseCycle and cycleRepetition.			
Base	ARObject,CommunicationCycle			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
BaseCycle	Integer	1	attr	The first communication cycle where the frame is sent. This value is incremented at the beginning of each new cycle, ranging from 0 to 63, and is reset to 0 after a sequence of 64 cycles.
CycleRepetition	CycleRepetitionType	1	attr	The number of communication cycles (after the first cycle) whenever the frame described by this timing is sent again.

Table 5.50: CycleRepetition

Enumeration	**CycleRepetitionType**
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::FibexCore::CoreTopology
Note	The number of communication cycles (after the first cycle) whenever the frame is sent again. The FlexRay communication controller allows only determined values.
Literal	**Description**
cycleRepetition1	Attribute cycleRepetition value="1" valid only for FlexRay Protocol 2.1 Rev A
cycleRepetition10	Attribute cycleRepetition value="10" to support FlexRay 3.0
cycleRepetition16	Attribute cycleRepetition value="16" valid only for FlexRay Protocol 2.1 Rev A
cycleRepetition2	Attribute cycleRepetition value="2" valid only for FlexRay Protocol 2.1 Rev A
cycleRepetition20	Attribute cycleRepetition value="20" to support FlexRay 3.0
cycleRepetition32	Attribute cycleRepetition value="32" valid only for FlexRay Protocol 2.1 Rev A
cycleRepetition4	Attribute cycleRepetition value="4" valid only for FlexRay Protocol 2.1 Rev A
cycleRepetition40	Attribute cycleRepetition value="40" to support FlexRay 3.0
cycleRepetition5	Attribute cycleRepetition value="5" to support FlexRay 3.0
cycleRepetition50	Attribute cycleRepetition value="50" to support FlexRay 3.0
cycleRepetition64	Attribute cycleRepetition value="64" valid only for FlexRay Protocol 2.1 Rev A
cycleRepetition8	Attribute cycleRepetition value="8" valid only for FlexRay Protocol 2.1 Rev A

Table 5.51: CycleRepetitionType

5.9 LIN specific description

LIN is a protocol that is based on a single master - multiple slave principle. In the following, the parameters will be specified, which are necessary to describe the LIN Schedule Tables and the LIN Frames.

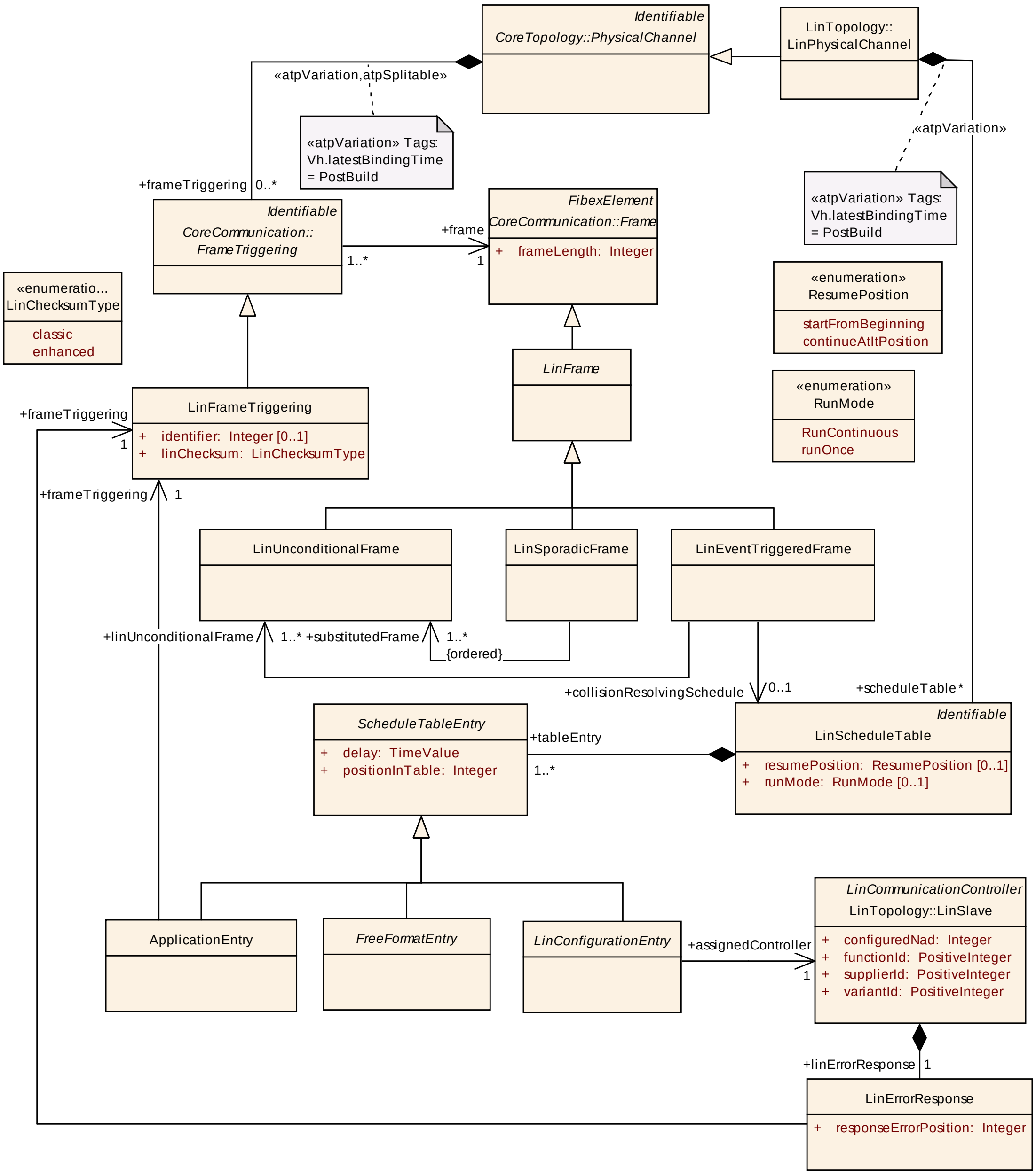

Figure 5.14: LIN Schedule Table (Fibex4Lin:LinScheduleTable)

5.9.1 LIN Frames

One LIN Frame consists of the two parts header and response. The header is always sent by the LIN Master, while the response is sent by only one dedicated LIN-Slave.

There are three different ways of transmitting frames on the bus: unconditional, event triggered, and sporadic frames.

Class	**LinFrame (abstract)**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Lin::LinCommunication			
Note	Lin specific Frame element.			
Base	ARObject,CollectableElement,FibexElement,Frame,Identifiable,Multilanguage Referrable,PackageableElement,Referrable			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
–	–	–	–	–

Table 5.52: LinFrame

Class	**LinFrameTriggering**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Lin::LinCommunication			
Note	LIN specific attributes to the FrameTriggering			
Base	ARObject,FrameTriggering,Identifiable,MultilanguageReferrable,Referrable			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
identifier	Integer	0..1	attr	To describe a frames identifier on the communication system, usually with a fixed identifierValue. For LinSporadicFrames the attribute shall be ignored.
linChecksum	LinChecksumType	1	attr	Type of checksum that the frame is using.

Table 5.53: LinFrameTriggering

Enumeration	**LinChecksumType**
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Lin::LinCommunication
Note	Use of classic or enhanced checksum is managed by the master node and it is determined per frame identifier;
Literal	***Description***
classic	Classic in communication with LIN 1.3 slave nodes
enhanced	Enhanced in communication with LIN 2.0 slave nodes.

Table 5.54: LinChecksumType

Class	LinUnconditionalFrame			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Lin::LinCommunication			
Note	Unconditional frames carry signals. The master sends a frame header in a scheduled frame slot and the designated slave node fills the frame with data. **Tags:** atp.recommendedPackage=Frames			
Base	ARObject,CollectableElement,FibexElement,Frame,Identifiable,Lin Frame,MultilanguageReferrable,PackageableElement,Referrable			
Attribute	Datatype	Mul.	Kind	Note
–	–	–	–	–

Table 5.55: LinUnconditionalFrame

Class	LinSporadicFrame			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Lin::LinCommunication			
Note	A sporadic frame is a group of unconditional frames that share the same frame slot. The sporadic frame shall not contain any Pdus. **Tags:** atp.recommendedPackage=Frames			
Base	ARObject,CollectableElement,FibexElement,Frame,Identifiable,Lin Frame,MultilanguageReferrable,PackageableElement,Referrable			
Attribute	Datatype	Mul.	Kind	Note
substitutedFrame (ordered)	LinUnconditionalFrame	1..*	ref	Reference to a group of unconditional frames that share the same frame slot. In case that more than one of the declared frames needs to be transferred, the one first listed shall be chosen. Within a channel a LIN Frame shall be referenced by only one FrameTriggering. This allows a derivation of the identifier of a substituted Frame. The identifier is specified in FrameTriggering element. A LinUnconditionalFrame associated with a LinSporadicFrame may not be allocated in the same LinScheduleTable as the sporadic frame.

Table 5.56: LinSporadicFrame

Class	LinEventTriggeredFrame			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Lin::LinCommunication			
Note	An event triggered frame is used as a placeholder to allow multiple slave nodes to provide its response. The header of an event triggered frame is transmitted when a frame slot allocated to the event triggered frame is processed. The publisher of an associated unconditional frame shall only transmit the response if at least one of the signals carried in its unconditional frame is updated. The LIN Master discovers and purges collisions with the collisionResolvingScheduleTable. The event controlled frame shall not contain any Pdus. **Tags:** atp.recommendedPackage=Frames			
Base	ARObject,CollectableElement,FibexElement,Frame,Identifiable,LinFrame,MultilanguageReferrable,PackageableElement,Referrable			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**
collisionResolvingSchedule	LinScheduleTable	0..1	ref	Reference to the schedule table, which resolves a collision.
linUnconditionalFrame	LinUnconditionalFrame	1..*	ref	A list of slaves can respond to the master request if at least one of the signals carried in its unconditional frame is updated. For each response a LinFrameTriggering and a LinUnconditionalFrame shall be defined. Within a channel a LIN Frame shall be referenced by only one FrameTriggering. This allows a derivation of the identifier of a substituted Frame. The identifier is specified in FrameTriggering element. The Unconditional frames associated with an event triggered frame shall: • have equal length. • use the same checksum model (i.e. mixing LIN 1.x and LIN 2.x frames is not allowed). • reserve the first data field to its protected identifier (even if the associated unconditional frame is scheduled as a unconditional frame in the same or another schedule table). • be published by different slave nodes. • shall not be included directly in the same schedule table as the event triggered frame is scheduled.

Table 5.57: LinEventTriggeredFrame

5.9.2 LIN Schedule Table

The LIN-Master uses one or more predefined scheduling tables to start the sending and receiving to the LIN bus. These scheduling tables contains at least the relative timing, where the message sending is initiated.

Class	**LinScheduleTable**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Lin::LinCommunication			
Note	The master task (in the master node) transmits frame headers based on a schedule table. The schedule table specifies the identifiers for each header and the interval between the start of a frame and the start of the following frame.			
Base	ARObject,Identifiable,MultilanguageReferrable,Referrable			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
resumePosition	ResumePosition	0..1	attr	Defines, where a schedule table shall be proceeded in case if it has been interrupted by a run-once table or MRF/SRF.
runMode	RunMode	0..1	attr	The schedule table can be executed in two different modes.
tableEntry	ScheduleTableEntry	1..*	aggr	The scheduling table consists of table entries, which contain Frame slots.

Table 5.58: LinScheduleTable

Enumeration	**RunMode**
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Lin::LinCommunication
Note	The schedule table can be executed in two different modes.
Literal	**Description**
RunContinuous	RUN_CONTINUOUS run mode
runOnce	RUN_ONCE run mode

Table 5.59: RunMode

Enumeration	**ResumePosition**
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Lin::LinCommunication
Note	Defines, where a schedule table shall be proceeded in case if it has been interrupted by a run-once table or MRF/SRF.
Literal	**Description**
continueAtItPosition	Continue at IT Point.
startFromBeginning	Start from the beginning

Table 5.60: ResumePosition

Class	**ScheduleTableEntry (abstract)**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Lin::LinCommunication			
Note	Table entry in a LinScheduleTable. Specifies what will be done in the frame slot.			
Base	ARObject			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**
delay	TimeValue	1	attr	Relative delay between this tableEntry and the start of the successor in the schedule table in seconds.
introduction	DocumentationBlock	0..1	aggr	This represents introductory documentation about the schedule table entry. **Tags:** xml.sequenceOffset=-10
positionInTable	Integer	1	attr	Relative position in the schedule table. The first entry index in the schedule table is 0.

Table 5.61: ScheduleTableEntry

Class	**ApplicationEntry**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Lin::LinCommunication			
Note	There are three types of frames defined in LIN, i.e. unconditional frame, event triggered frame and sporadic frame. The ApplicationEntry refers to a frame that will be transfered in the frame slot.			
Base	ARObject,ScheduleTableEntry			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**

Attribute	Datatype	Mul.	Kind	Note
frameTriggering	LinFrameTriggering	1	ref	Specifies the LinFrame that will be transmitted in this frame slot.

Table 5.62: ApplicationEntry

Class	FreeFormatEntry (abstract)			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Lin::LinCommunication			
Note	FreeFormat transmits a fixed master request frame with the eight data bytes provided. This may for instance be used to issue user specific fixed frames.			
Base	ARObject,ScheduleTableEntry			
Attribute	Datatype	Mul.	Kind	Note
–	–	–	–	–

Table 5.63: FreeFormatEntry

Class	LinConfigurationEntry (abstract)			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Lin::LinCommunication			
Note	A ScheduleTableEntry which contains LIN specific assignments.			
Base	ARObject,ScheduleTableEntry			
Attribute	Datatype	Mul.	Kind	Note
assignedController	LinSlave	1	ref	The LIN slaves controller who is target of this assignment.

Table 5.64: LinConfigurationEntry

5.9.3 Configuration Services

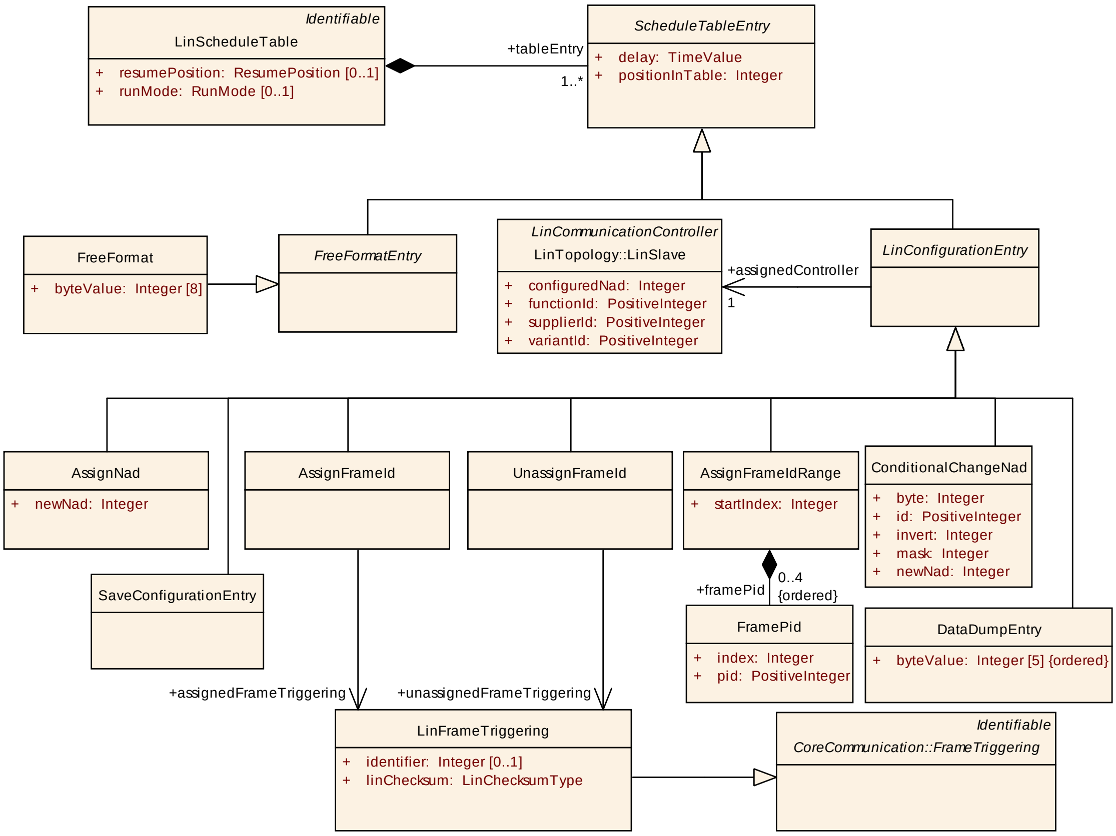

Figure 5.15: LIN Configuration Entries (Fibex4Lin:LinConfigurationEntries)

LIN only supports 64 identifiers. That creates the need for extending the address space. Hence the frames are identified by message ids from a much larger address space that is additionally separated by supplier ids. During runtime the master assigns a LinId to the frame. In case of identical parts within a cluster the initial node ID (oldNad) is used to differentiate such nodes.

To support that in System Template the `AssignFrameId` is introduced as a LIN specific extension. For the assignment a relation to the `LinSlave` is used. The `LinSlave` element is referenced by a `LINCommunicationConnector` element that contains a list of frames processed by the slave node. More details can be found in chapter 5.9.3.

Class	**AssignFrameId**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Lin::LinCommunication			
Note	Schedule entry for an Assign Frame Id master request.			
Base	ARObject,LinConfigurationEntry,ScheduleTableEntry			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**
assignedFrameTriggering	LinFrameTriggering	1	ref	The frame whose identifier is set by this assignment.

Table 5.65: AssignFrameId

Class	**UnassignFrameId**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Lin::LinCommunication			
Note	Schedule entry for an Unassign Frame Id master request where the protected identifier is assigned the value 0x40. This will disable reception/transmission of a previously dynamically assigned frame identifier.			
Base	ARObject,LinConfigurationEntry,ScheduleTableEntry			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
unassignedFrameTriggering	LinFrameTriggering	1	ref	The frame whose identifier is reset by this assignment.

Table 5.66: UnassignFrameId

The Assign frame ID configuration service is replaced in LIN 2.1 by the Assign frame ID range configuration service. AssignmentFrameIdRange is used to set or disable Protected Identifiers up to four frames. For the assignment a relation to the LinSlave is used. The LinSlave element is referenced by a LINCommunicationConnector element that contains a list of frames processed by the slave node. More details can be found in chapter 5.9.3.

Class	**AssignFrameIdRange**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Lin::LinCommunication			
Note	AssignFrameIdRange generates an assign frame PID range request.			
Base	ARObject,LinConfigurationEntry,ScheduleTableEntry			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
framePid (ordered)	FramePid	0..4	aggr	Optional assignment of frame_PID values that are included in the request. The frame_PIDs are ordered.
startIndex	Integer	1	attr	The startIndex sets the index to the first frame to assign a PID.

Table 5.67: AssignFrameIdRange

Class	**FramePid**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Lin::LinCommunication			
Note	Frame_PIDs that are included in the request. The "pid" attribute describes the value and the "index" attribute the position of the frame_PID in the request.			
Base	ARObject			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
index	Integer	1	attr	This attribute is used to order the frame_PIDs. The values of index shall be unique within one AssignFrameIdRange.
pid	PositiveInteger	1	attr	Frame_PID value.

Table 5.68: FramePid

Assign NAD is used to resolve conflicting NADs in LIN clusters built using off-the-shelves slave nodes or reused slave nodes. This request uses the initial NAD. The NAD used for the response shall be the same as in the request, i.e. the initial NAD.

Class	**AssignNad**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Lin::LinCommunication			
Note	Schedule entry for an Assign NAD master request.			
Base	ARObject,LinConfigurationEntry,ScheduleTableEntry			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
newNad	Integer	1	attr	The newly assigned NAD value.

Table 5.69: AssignNad

— AUTOSAR CONFIDENTIAL —

The conditional change NAD is used to detect unknown slave nodes in a cluster and to separate their NADs.

Class	**ConditionalChangeNad**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Lin::LinCommunication			
Note	Generates an conditional change NAD request. See LIN 2.1 protocol specification for more information.			
Base	ARObject,LinConfigurationEntry,ScheduleTableEntry			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
byte	Integer	1	attr	Byte Position of Data Byte that should be used for the bitwise XOR with Invert and the bitwise AND with Mask.
id	PositiveInteger	1	attr	Byte Position of Id.
invert	Integer	1	attr	Byte Position of Invert.
mask	Integer	1	attr	Byte Position of Mask.
newNad	Integer	1	attr	The newly assigned NAD value (Byte Position).

Table 5.70: ConditionalChangeNad

The Save Configuration service tells the slave node that the slave application shall save the current configuration.

Class	**SaveConfigurationEntry**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Lin::LinCommunication			
Note	This service is used to notify a slave node to store its configuration.			
Base	ARObject,LinConfigurationEntry,ScheduleTableEntry			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
–	–	–	–	–

Table 5.71: SaveConfigurationEntry

The Data Dump service is reserved for initial configuration of a slave node by the slave node supplier and the format of this message is supplier specific.

Class	**DataDumpEntry**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Lin::LinCommunication			
Note	This service is reserved for initial configuration of a slave node by the slave node supplier and the format of this message is supplier specific.			
Base	ARObject,LinConfigurationEntry,ScheduleTableEntry			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
byteValue (ordered)	Integer	5	attr	Supplier specific format.

Table 5.72: DataDumpEntry

With the FreeFormat a scheduling of fixed data content within a diagnostic frame is defined. For that specification `FreeFormat` is introduced.

Class	**FreeFormat**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Lin::LinCommunication			
Note	Representing freely defined data.			
Base	ARObject,FreeFormatEntry,ScheduleTableEntry			
Attribute	*Datatype*	*Mul.*	*Kind*	*Note*
byteValue	Integer	8	attr	The integer Value of a freely defined data byte.

Table 5.73: FreeFormat

In order to be consistent with the rest of the communication configuration, it is required that the diagnostic LIN Frames (Master Request Frame, Slave Response Frame) are explicitly modeled as `Frame` elements. `LinFrameTriggerings` dealing with diagnostic Frames thus reference this diagnostic frames.

5.10 CAN specific description

This chapter describes additions to the CAN definition of Frames.

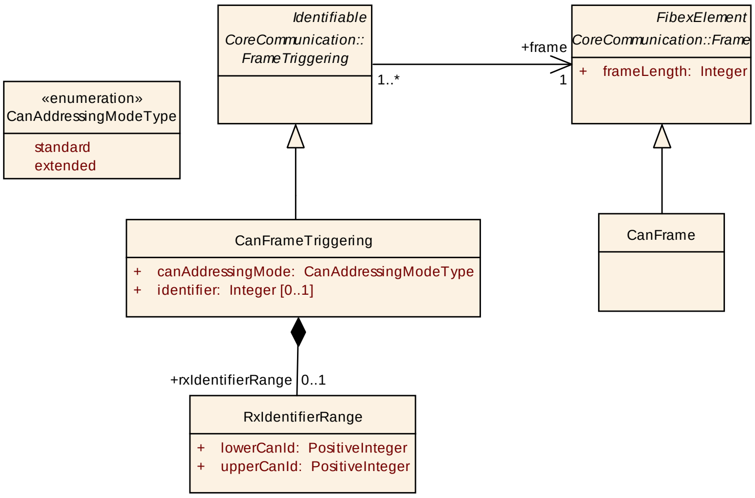

Figure 5.16: CanFrameTriggering (Fibex4Can:CanCommunication)

Class	**CanFrame**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Can::CanCommunication			
Note	CAN specific Frame element. This element shall also be used for TTCan. **Tags:** atp.recommendedPackage=Frames			
Base	ARObject,CollectableElement,FibexElement,Frame,Identifiable,Multilanguage Referrable,PackageableElement,Referrable			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
–	–	–	–	–

Table 5.74: CanFrame

Class	**CanFrameTriggering**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Can::CanCommunication			
Note	CAN specific attributes to the FrameTriggering			
Base	ARObject,FrameTriggering,Identifiable,MultilanguageReferrable,Referrable			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
absolutely Scheduled Timing	TtcanAbsolutely ScheduledTimin g	*	aggr	Each frame in TTCAN is identified by its slot id and communication cycle. A description is provided by the usage of AbsolutelyScheduledTiming.
canAddres singMode	CanAddressing ModeType	1	attr	The CAN protocol supports two types of frame formats. The standard frame format uses 11-bit identifiers and is defined in the CAN specification 2.0 A. Additionally the extended frame format allows 29-bit identifiers and is defined in the CAN specification 2.0 B.

Attribute	Datatype	Mul.	Kind	Note
identifier	Integer	0..1	attr	To describe a frames identifier on the communication system, usually with a fixed identifierValue. In a complete system description this attribute is mandatory. In an Ecu Extract for the sender of the frame the identifier shall also be provided. In an Ecu Extract for the receiver the identifier attribute shall be ignored if rxIdentifierRange is defined.
rxIdentifier Range	RxIdentifierRan ge	0..1	aggr	Optional definition of a CanId range.

Table 5.75: CanFrameTriggering

Enumeration	CanAddressingModeType
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Can::Can Communication
Note	Indicates whether standard or extended CAN identifiers are used
Literal	**Description**
extended	Extended 29-bit-identifiers are used (CAN 2.0B)
standard	Standard 11-bit-identifiers are used (CAN 2.0A)

Table 5.76: CanAddressingModeType

Class	RxIdentifierRange			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Can::CanCommunication			
Note	Optional definition of a CanId range to reduce the effort of specifying every possible FrameTriggering within the defined Id range during reception. All frames received within a range are mapped to the same Pdu that is passed to a upper layer module (e.g. Nm, CDD, PduR). This range is redundant to the nmRangeConfig attributes of "CanNmNode". For backward compatibility reasons this redundancy shall be preserved and both shall be defined.			
Base	ARObject			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**
lowerCanI d	PositiveInteger	1	attr	This attribute can be used together with the upperCanId attribute to define a range of CanIds.
upperCanI d	PositiveInteger	1	attr	This attribute can be used together with the lowerCanId attribute to define a range of CanIds.

Table 5.77: RxIdentifierRange

5.11 TTCAN specific description

This chapter describes additions to the TTCAN definition of Frames.

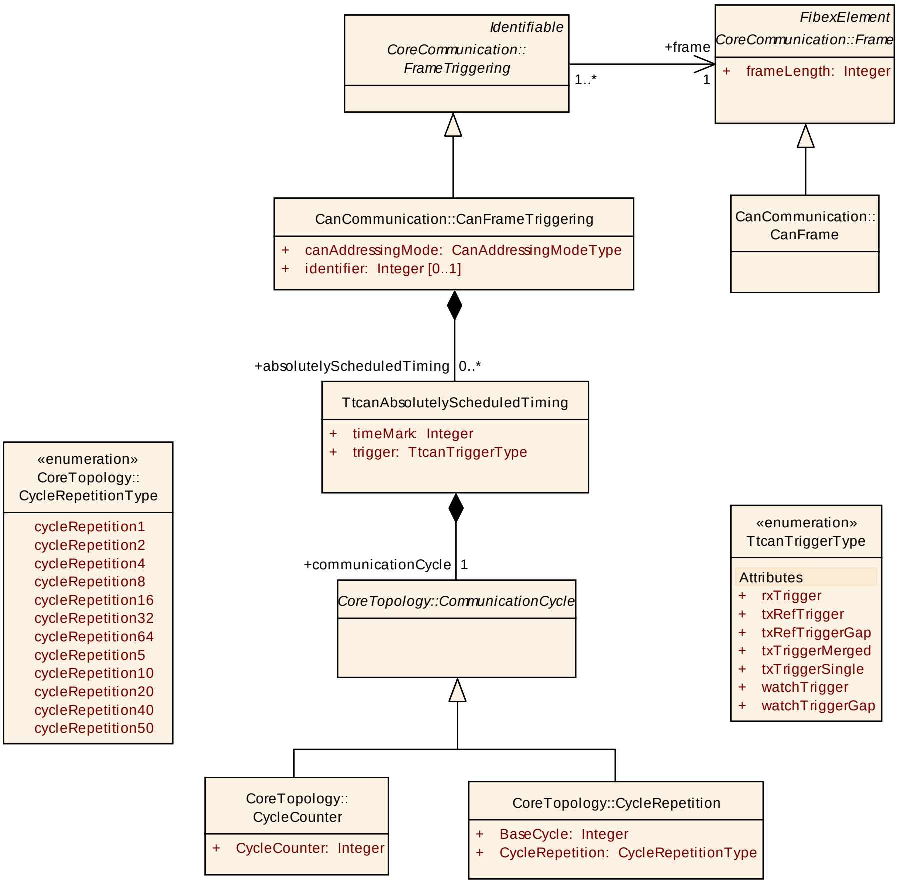

Figure 5.17: TtcanAbsolutelyScheduledTiming (Fibex4Ttcan:TtcanCommunication)

Class	**TtcanAbsolutelyScheduledTiming**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Ttcan::Ttcan Communication			
Note	Each frame in TTCAN is identified by its slot id and communication cycle. A description is provided by the usage of AbsolutelyScheduledTiming. A frame can be sent multiple times within one communication cycle. For describing this case multiple AbsolutelyScheduledTimings have to be used. The main use case would be that a frame is sent twice within one communication cycle.			
Base	ARObject			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**
communicationCycle	CommunicationCycle	1	aggr	The communication cycle where the frame is sent.
timeMark	Integer	1	attr	Where FlexRay counts the slots in the static segment, TTCAN requires explicit Tx and Rx time marks.
trigger	TtcanTriggerType	1	attr	Trigger type for this time window.

Table 5.78: TtcanAbsolutelyScheduledTiming

Enumeration	**TtcanTriggerType**
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Ttcan::Ttcan Communication
Note	This type lists all trigger types for a time window.
Literal	**Description**
rxTrigger	Check for message reception
txRefTrigger	Send reference message in periodic case
txRefTriggerGap	Send reference message in event-synchronised case
txTriggerMerged	Send message in a merged arbitration window
txTriggerSingle	Send message in an exclusive time window
watchTrigger	Check for missing reference message in periodic case
watchTriggerGap	Check for missing reference message in event-synchronised case

Table 5.79: TtcanTriggerType

5.12 Ethernet specific description

AUTOSAR supports TCP/IP and UDP/IP communication over Ethernet. This section specifies the information of the AUTOSAR Basic Software module Socket Adaptor (SoAd) that is common for several ECUs and therefore is part of the System Configuration Description. The main purpose of the SoAd module is to create an interface between the PDU Router and a socket based TCP/IP stack.

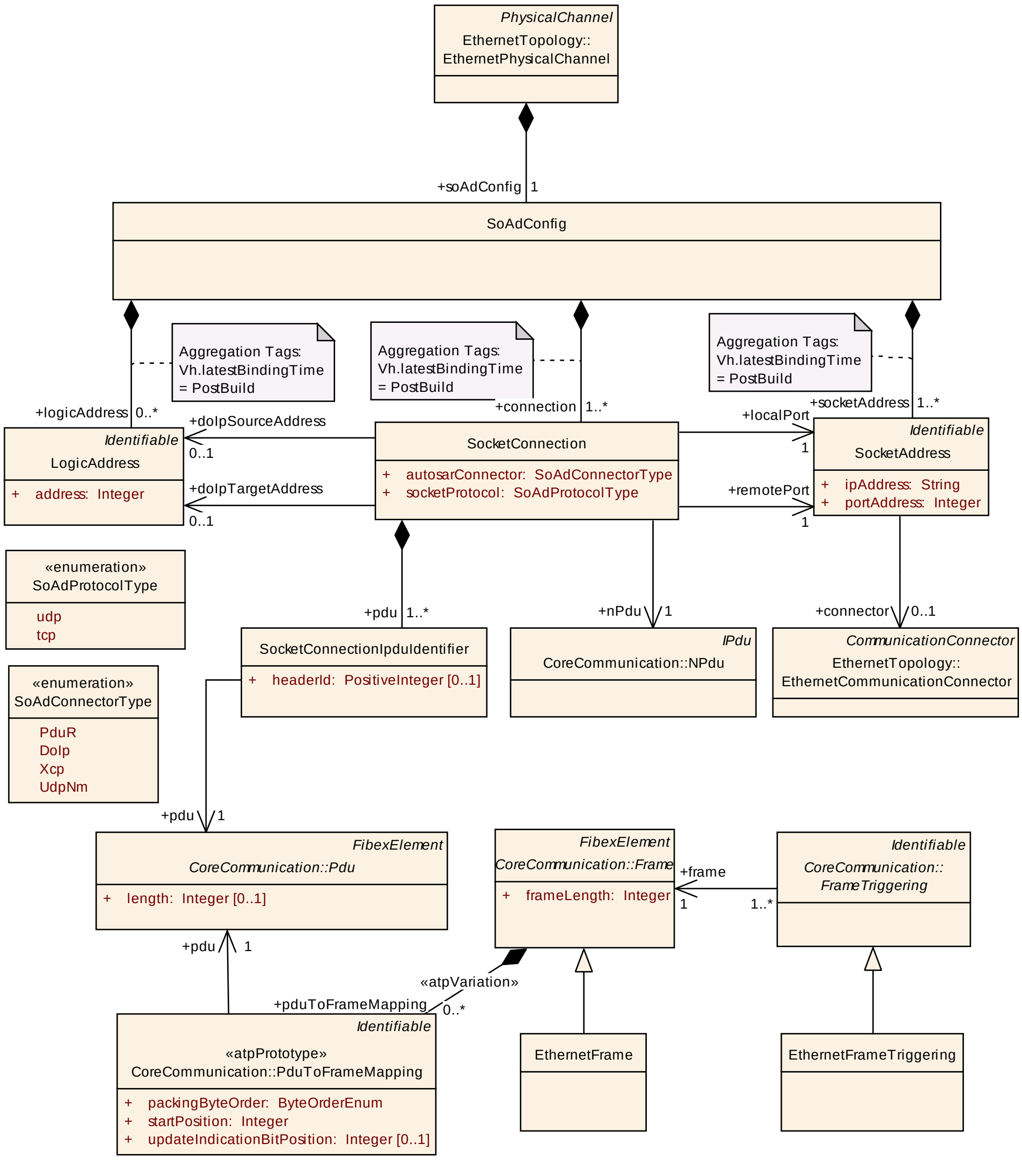

Figure 5.18: Ethernet Communication (Fibex4Ethernet:EthernetCommunication)

The SoAd serves as a (De)Multiplexer between different PDU sources/suppliers and the TCP/IP stack. The `SocketConnection` maps TCP/UDP Ports as well as IP addresses to the `IPdu` and add this information during transmit. On receive it needs to reverse this process and create the `IPdu` from the TCP/IP information received. The element `NPdu` is used to describe the datagram that is transmitted from the SoAd to the Ethernet Interface. Multiple `SocketConnections` can be defined on the same ECU.

Class	**EthernetFrame**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Ethernet::Ethernet Communication			
Note	Ethernet specific attributes to the Frame **Tags:** atp.recommendedPackage=Frames			
Base	ARObject,CollectableElement,FibexElement,Frame,Identifiable,Multilanguage Referrable,PackageableElement,Referrable			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**
–	–	–	–	–

Table 5.80: EthernetFrame

Class	**EthernetFrameTriggering**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Ethernet::Ethernet Communication			
Note	Ethernet specific Frame element.			
Base	ARObject,FrameTriggering,Identifiable,MultilanguageReferrable,Referrable			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**
–	–	–	–	–

Table 5.81: EthernetFrameTriggering

Class	**SoAdConfig**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Ethernet::Ethernet Communication			
Note	SoAd Configuration for one specific Physical Channel.			
Base	ARObject			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**
connection	SocketConnection	1..*	aggr	Collection of socket connections. **Tags:** Vh.latestBindingTime=PostBuild
logicAddress	LogicAddress	*	aggr	Collection of DoIp Addresses. **Tags:** Vh.latestBindingTime=PostBuild
socketAddress	SocketAddress	1..*	aggr	Collection of SoAdAddresses. **Tags:** Vh.latestBindingTime=PostBuild

Table 5.82: SoAdConfig

Class	SocketConnection			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Ethernet::Ethernet Communication			
Note	The SoAd serves as a (De)Multiplexer between different PDU sources and the TCP/IP stack.			
Base	ARObject			
Attribute	Datatype	Mul.	Kind	Note
autosarConnector	SoAdConnectorType	1	attr	Availability of protocol plug-ins. Entries in the Socket and PDU Routing Tables.
doIpSourceAddress	LogicAddress	0..1	ref	The logical DoIP address of the source entity. This optional reference shall only be used for DoIP (Diagnosis over IP).
doIpTargetAddress	LogicAddress	0..1	ref	The logical DoIP address of the target entity. This optional reference shall only be used for DoIP (Diagnosis over IP).
localPort	SocketAddress	1	ref	Local Port for TCP/UDP connection (in case the local port is fixed).
nPdu	NPdu	1	ref	Reference to data packets that are transmitted over Ethernet. Each data packet can contain multiple IPdus.
pdu	SocketConnectionIpduIdentifier	1..*	aggr	PDUs handed over by the PDU Router (Transmission over the Ethernet) or PDUs handed over by SoAd (Reception over Ethernet). Multiple IPdus can be transmitted over one socket connection.
remotePort	SocketAddress	1	ref	Remote Port for TCP/UDP connection. May be different for each Frame or use the same remote port. In second case headerId attribute needs to be considered.
socketProtocol	SoAdProtocolType	1	attr	Specifies the transport protocol (UDP or TCP). Transport Protocols are responsible for encapsulating application data blocks into datagrams suitable for transfer to the network infrastructure for transmission to the destination host, or managing the reverse transaction by abstracting network datagrams and delivering them to an application.

Table 5.83: SocketConnection

Class	SocketAddress			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Ethernet::Ethernet Communication			
Note	SocketAddress contains the portAddress and the ipAddress.			
Base	ARObject,Identifiable,MultilanguageReferrable,Referrable			
Attribute	Datatype	Mul.	Kind	Note
connector	EthernetCommunicationConnector	0..1	ref	Asscociation to a CommunicationConnector in the topology description. In a System Description this reference is mandatory. In an ECU Extract this reference is optional (references to ECUs that are not part of the ECU Extract shall be avoided).

Attribute	Datatype	Mul.	Kind	Note
ipAddress	String	1	attr	Logical address that is assigned to the referenced device in a network utilizing the Internet Protocol for communication between its nodes.
portAddress	Integer	1	attr	Remote or Local UDP or TCP port used for the connection that refers this element.

Table 5.84: SocketAddress

Class	**SocketConnectionIpduIdentifier**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Ethernet::EthernetCommunication			
Note	An Identifier is required in case of one port per ECU communication where multiple Pdus are transmitted over the same connection. If only one IPdu is transmitted over the connetion this attribute can be ignored.			
Base	ARObject			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**
headerId	PositiveInteger	0..1	attr	If multiple Pdus are transmitted over the same connection this headerId can be used to distinguish between the different Pdus.
pdu	Pdu	1	ref	Reference to an IPdu that is mapped to a socket connection. This reference can be used to derive the AUTOSAR Connector in SoAd configuration.

Table 5.85: SocketConnectionIpduIdentifier

Class	**LogicAddress**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Ethernet::EthernetCommunication			
Note	The logical DoIP address. This element shall only be used for DoIP (Diagnosis over IP).			
Base	ARObject,Identifiable,MultilanguageReferrable,Referrable			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**
address	Integer	1	attr	The logical DoIP address.

Table 5.86: LogicAddress

Enumeration	**SoAdProtocolType**
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Ethernet::EthernetCommunication
Note	Transport Protocols above IP.
Literal	**Description**
tcp	Transmission Control Protocol (TCP) enables two hosts to establish a connection and exchange streams of data.
udp	User Datagram Protocol (UDP) offers a connectionless way to send and receive datagrams over an IP network. It's used primarily for broadcasting messages over a network.

Table 5.87: SoAdProtocolType

Enumeration	**SoAdConnectorType**
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Ethernet::Ethernet Communication
Note	Availability of protocol plug-ins. Entries in the Socket and PDU Routing Tables.
Literal	***Description***
DoIp	Diagnosis over IP
PduR	Pdu Router
UdpNm	Udp Nm
Xcp	Universal Measurement and Calibration Protocol

Table 5.88: SoAdConnectorType

5.12.1 Diagnostics over IP

In AUTOSAR the DoIp functionality is implemented in the same software module as the SoAd functionality. It will encapsulate, split and reassemble data and manage connections using the DoIP wrapper specification and thus has the functionality of a transport protocol similar to the CAN-TP or FlexRay-TP. It utilizes the socket interface of the TCP/IP stack module to send and receive UDP data packets and receive and transmit data streams through TCP-sockets. Therefore the model from figure 5.18 shall be used to describe DoIp in the System Description. The optional references to the `LogicAddress` are only relevant for the DoIp description.

5.13 SAE J1939 Protocol specific description

J1939 is a protocol and application layer standard of the SAE (Society of Automotive Engineers) based on the CAN technology. It defines parameters uniquely identified by the SPN (Suspect Parameter Number). These are mapped to parameter groups that are uniquely identified by a PGN (Parameter Group Number). Parameters are simply handled as `SystemSignals` which have a name derived from the name of the SPNs. A Parameter Group (PG) corresponds to an `IPdu`.

J1939 uses extended 29 bit CAN identifiers to encode a priority, the source address of the frame, and a frame ID which is based on the PGN (Parameter Group Number) and may contain the destination address.

The System Template does not introduce a new J1939 Communication Cluster, but rather J1939 Messages are handled within an ordinary CAN Communication Cluster, since a mixed traffic with CAN is allowed.

J1939 supports IPdus with more than 8 bytes, and IPdus with variable length that may exceed 8 bytes. As soon as an IPdu has more than 8 bytes, it does not fit in a single CAN frame and a transport protocol must be used. Variable length IPdus will always be handled by the J1939 TP, regardless of the actual length. The J1939 Transport Protocol is described in chapter 5.15.5.

5.14 I-Pdu Timing

AUTOSAR COM allows configuring statically two different transmission modes for each IPdu (True and False). `TransmissionModeDeclaration` uses a transmission mode selector, calculated from a number of individual `TransmissionModeConditions` or `ModeDrivenTransmissionModeConditions` to decide which of the two modes is selected. It is possible to switch between the transmission modes during runtime.

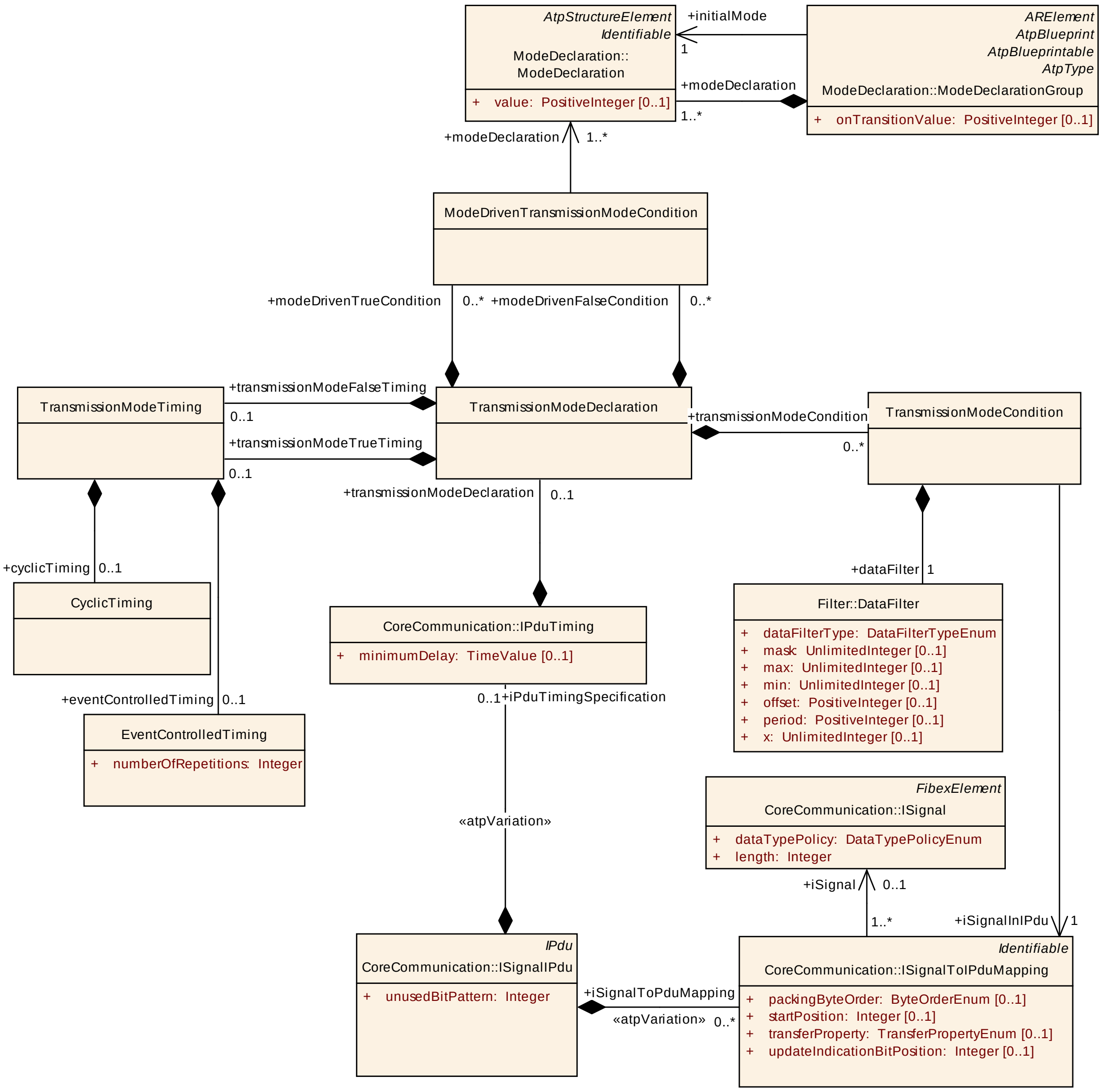

Figure 5.19: IPdu Timing

As Transmission mode selector the signal content can be evaluated via `transmissionModeCondition` or mode conditions can be defined with the `modeDrivenTrueCondition` or `modeDrivenFalseCondition`. The mode evaluation and the signal content evaluation shall not be used in the same IPdu. A mix of these two types is not allowed.

To use the signal content evaluation a TransmissionModeCondition can be attached to each signal within an I-PDU. Each TransmissionModeCondition contains a reference to a signal and to an assigned filter. The filter condition is used for the selection of the transmission mode. If at least one condition in the signal content evaluation is true, Transmission Mode "TRUE" shall be used for this I-Pdu. In all other cases, the Transmission Mode "FALSE" shall be used. More details can be found in the COM Specification [19].

In the mode driven evaluation ModeDeclarations are evaluated. The transmissionModeFalseTiming is activated if all defined modeDrivenFalseConditions evaluate to true and the transmissionModeTrueTiming is activated if all defined modeDrivenTrueConditions evaluate to true. Each condition that is defined by ModeDrivenTransmissionModeCondition evaluates to true if one of the referenced ModeDeclarations is active.

COM Transmission Modes	*Description*	*realization in System Template*
Periodic	Transmissions occur indefinitely with a fixed period between them	CyclicTiming
Direct/n-times	Event driven transmission with n-1 repetitions	EventControlledTiming
Mixed	Periodic transmission with direct/n-times transmissions in between	EventControlledTiming and CyclicTiming
None	No transmission	no timing assigned

Table 5.89: COM Transmission Modes

The TransmissionModeDeclaration element aggregates the TransmissionModeTiming in two different roles: transmissionModeTrueTiming and transmissionModeFalseTiming. The available COM Transmission Mode Timings can be described by the CyclicTiming and EventControlledTiming elements (see Table 5.89) that are aggregated by the TransmissionModeTiming class.

Class	**TransmissionModeDeclaration**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::FibexCore::CoreCommunication::Timing			
Note	AUTOSAR COM provides the possibility to define two different TRANSMISSION MODES (True and False) for each I-PDU. As TransmissionMode selector the signal content can be evaluated via transmissionModeCondition (implemented directly in the COM module) or mode conditions can be defined with the modeDrivenTrueCondition or modeDrivenFalseCondition (evaluated by BswM and invoking Com_SwitchIpduTxMode COM API). If modeDrivenTrueCondition and modeDrivenFalseCondition are defined they shall never evaluate to true both at the same time. The mixing of Transmission Mode Switch via API and signal value is not allowed.			
Base	ARObject			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**
modeDrivenFalseCondition	ModeDrivenTransmissionModeCondition	*	aggr	Defines the trigger for the Com_SwitchIpduTxMode Transmission Mode switch. Only if all defined modeDrivenFalseConditions evaluate to true (AND associated) the transmissionModeFalseTiming shall be activated. modeDrivenTrueCondition and modeDrivenFalseCondition shall never evaluate to true both at the same time.
modeDrivenTrueCondition	ModeDrivenTransmissionModeCondition	*	aggr	Defines the trigger for the Com_SwitchIpduTxMode Transmission Mode switch. Only if all defined modeDrivenTrueConditions evaluate to true (AND associated) the transmissionModeTrueTiming shall be activated. modeDrivenTrueCondition and modeDrivenFalseCondition shall never evaluate to true both at the same time.
transmissionModeCondition	TransmissionModeCondition	*	aggr	The Transmission Mode Selector evaluates the conditions for a subset of signals and decides which transmission mode should be used. In case only one transmission mode is used there is no need for the "TransmissionModeCondition" and its sub-structure. In case the transmission mode shall be switched using the COM-API "Com_SwitchIpduTxMode" there is no need for the´"TransmissionModeCondition" and its sub-structure.
transmissionModeFalseTiming	TransmissionModeTiming	0..1	aggr	Timing Specification if the COM Transmission Mode is false. The Transmission Mode Selector is defined to be false, if all Conditions evaluate to false.
transmissionModeTrueTiming	TransmissionModeTiming	0..1	aggr	Timing Specification if the COM Transmission Mode is true. The Transmission Mode Selector is defined to be true, if at least one Condition evaluates to true.

Table 5.90: TransmissionModeDeclaration

Class	**TransmissionModeCondition**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::FibexCore::CoreCommunication::Timing			
Note	Possibility to attach a condition to each signal within an I-PDU. If at least one condition evaluates to true, TRANSMISSION MODE True shall be used for this I-Pdu. In all other cases, the TRANSMISSION MODE FALSE shall be used.			
Base	ARObject			
Attribute	*Datatype*	*Mul.*	*Kind*	*Note*
dataFilter	DataFilter	1	aggr	Possibilities to define conditions
iSignalInIPdu	ISignalToIPduMapping	1	ref	Reference to a signal to which a condition is attached.

Table 5.91: TransmissionModeCondition

Class	**ModeDrivenTransmissionModeCondition**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::FibexCore::CoreCommunication::Timing			
Note	The condition defined by this class evaluates to true if one of the referenced modeDeclarations (OR associated) is active. All referenced modeDeclarations shall be from the same ModeDeclarationGroup. The condition is used to define which TransmissionMode shall be activated using Com_SwitchIpduTxMode.			
Base	ARObject			
Attribute	*Datatype*	*Mul.*	*Kind*	*Note*
modeDeclaration	ModeDeclaration	1..*	ref	Reference to one modeDeclaration which is OR associated in the context of the ModeDrivenTransmissionModeCondition.

Table 5.92: ModeDrivenTransmissionModeCondition

The `ModeDeclaration` and the `ModeDeclarationGroup` is described in more detail in the Software Component Template Specification [4].

Class	**TransmissionModeTiming**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::FibexCore::CoreCommunication::Timing			
Note	If the COM Transmission Mode is false the timing is aggregated by the TransmissionModeTiming element in the role of transmissionModeFalseTiming. If the COM Transmission Mode is true the timing is aggregated by the TransmissionModeTiming element in the role of transmissionModeTrueTiming. COM supports the following Transmission Modes: Periodic (Cyclic Timing) Direct /n-times (EventControlledTiming) Mixed (Cyclic and EventControlledTiming are assigned) None (no timing is assigned)			
Base	ARObject			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**
cyclicTiming	CyclicTiming	0..1	aggr	Periodic Transmission Mode.
eventControlledTiming	EventControlledTiming	0..1	aggr	Direct Transmission Mode.

Table 5.93: TransmissionModeTiming

5.14.1 Data Filter configuration

Data Filters are used on sender side to configure Transmission Mode Conditions (TMC). On receiver side Data Filters can be used as filtering mechanisms for signals (see `ISignalPort` element). More details about the usage of `DataFilters` can be found in the Software Component Template Specification [4].

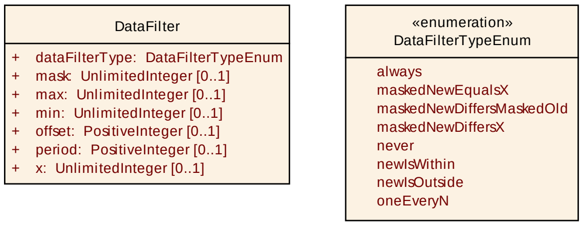

Figure 5.20: Data Filter

Class	**DataFilter**			
Package	M2::AUTOSARTemplates::CommonStructure::Filter			
Note	Base class for data filters. The type of the filter is specified in attribute dataFilterType. Some of the filter types require additional arguments which are specified as attributes of this class.			
Base	ARObject			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
dataFilterType	DataFilterTypeEnum	1	attr	This attribute specifies the type of the filter.
mask	UnlimitedInteger	0..1	attr	Mask for old and new value
max	UnlimitedInteger	0..1	attr	Value to specify the upper boundary
min	UnlimitedInteger	0..1	attr	Value to specify the lower boundary
offset	PositiveInteger	0..1	attr	Specifies the initial number of messages to occur before the first message is passed
period	PositiveInteger	0..1	attr	Specifies number of messages to occur before the message is passed again
x	UnlimitedInteger	0..1	attr	Value to compare with

Table 5.94: DataFilter

Enumeration	**DataFilterTypeEnum**
Package	M2::AUTOSARTemplates::CommonStructure::Filter
Note	This enum specifies the supported DataFilterTypes.
Literal	***Description***
always	No filtering is performed so that the message always passes.
maskedNewDiffersMaskedOld	Pass messages where the masked value has changed. (new_value&mask) !=(old_value&mask) new_value: current value of the message old_value: last value of the message (initialized with the initial value of the message, updated with new_value if the new message value is not filtered out)
maskedNewDiffersX	Pass messages whose masked value is not equal to a specific value x (new_value&mask) != x new_value: current value of the message

maskedNewEqualsX	Pass messages whose masked value is equal to a specific value x (new_value&mask) == x new_value: current value of the message
never	The filter removes all messages.
newIsOutside	Pass a message if its value is outside a predefined boundary. (min > new_value) OR (new_value > max)
newIsWithin	Pass a message if its value is within a predefined boundary. min <= new_value <= max
oneEveryN	Pass a message once every N message occurrences. Algorithm: occurrence % period == offset Start: occurrence = 0. Each time the message is received or transmitted, occurrence is incremented by 1 after filtering. Length of occurrence is 8 bit (minimum).

Table 5.95: DataFilterTypeEnum

— AUTOSAR CONFIDENTIAL —

5.14.2 Cyclic Timing

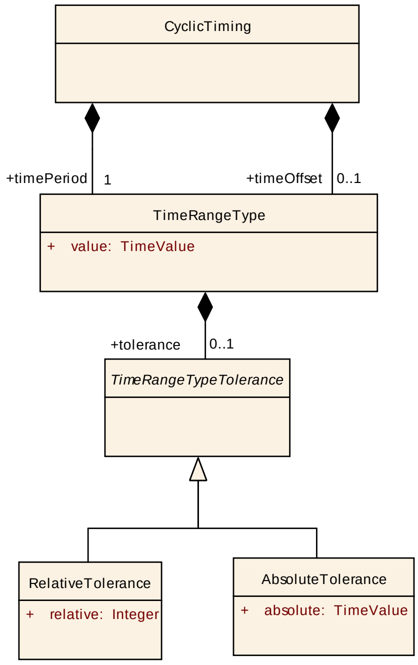

Figure 5.21: Cyclic Timing

Class	CyclicTiming			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::FibexCore::CoreCommunication::Timing			
Note	Specification of a cyclic sending behavior.			
Base	ARObject			
Attribute	Datatype	Mul.	Kind	Note
timeOffset	TimeRangeType	0..1	aggr	This attribute specifies the time until first transmission of this I-PDU. This attribute defines the time between Com_IpduGroupStart and the first transmission of the cyclic part of this transmission request for this I-PDU.
timePeriod	TimeRangeType	1	aggr	Period of the repetition of cyclic transmissions.

Table 5.96: CyclicTiming

5.14.3 EventControlled Timing

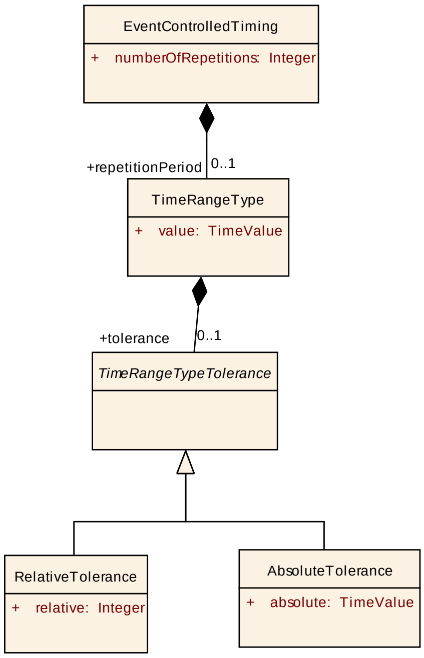

Figure 5.22: EventControlled Timing

Class	EventControlledTiming			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::FibexCore::CoreCommunication::Timing			
Note	Specification of a event driven sending behavior. The PDU is sent n (numberOfRepeat + 1) times separated by the repetitionPeriod. If numberOfRepeats = 0, then the Pdu is sent just once.			
Base	ARObject			
Attribute	Datatype	Mul.	Kind	Note

Attribute	Datatype	Mul.	Kind	Note
numberOf Repetitions	Integer	1	attr	Defines the number of repetitions for the Direct/N-Times transmission mode and the event driven part of Mixed transmission mode.
repetitionP eriod	TimeRangeTyp e	0..1	aggr	The repetitionPeriod specifies the time in seconds that elapses before the pdu can be sent the next time (Minimum repeat gap between two pdus). The repetitionPeriod is optional in case that no repetitions are configured.

Table 5.97: EventControlledTiming

Class	TimeRangeType			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::FibexCore::CoreCommunication:: Timing			
Note	The timeRange can be specified with the value attribute. Optionally a tolerance can be defined.			
Base	ARObject			
Attribute	Datatype	Mul.	Kind	Note
tolerance	TimeRangeTyp eTolerance	0..1	aggr	Optional specification of a tolerance.
value	TimeValue	1	attr	Average value of a date (in seconds)

Table 5.98: TimeRangeType

Class	RelativeTolerance			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::FibexCore::CoreCommunication:: Timing			
Note	Maximum allowable deviation			
Base	ARObject,TimeRangeTypeTolerance			
Attribute	Datatype	Mul.	Kind	Note
relative	Integer	1	attr	Maximum allowable deviation in percent

Table 5.99: RelativeTolerance

Class	AbsoluteTolerance			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::FibexCore::CoreCommunication:: Timing			
Note	Maximum allowable deviation			
Base	ARObject,TimeRangeTypeTolerance			
Attribute	Datatype	Mul.	Kind	Note
absolute	TimeValue	1	attr	Maximum allowable deviation in duration (in seconds)

Table 5.100: AbsoluteTolerance

5.15 Transport Layer

In AUTOSAR, the Transport Layer has two main purposes: The segmentation and reassembly of messages that are too long to fit into one frame on the underlying communication cluster, and the re-use of fixed frame identifiers for different message content.

According to the AUTOSAR Layered Software Architecture [13], each type of communication cluster has its own definition of the Transport Layer. Consequently, the peculiarities of the cluster types are addressed in the System Template by having different detailed models for FlexRay, CAN, LIN and J1939. However, all models are embedded into the communication model: They use specialized classes of TpConfig as a root element into the TP configuration. A TpConfig element is existing always in the context of exactly one CommunicationCluster. All Transport Layers will take IPdus as input elements, which will be transferred in the form of one or more NPdus. A TpConnection identifies a connection link between different communication nodes and routes the Pdus between them.

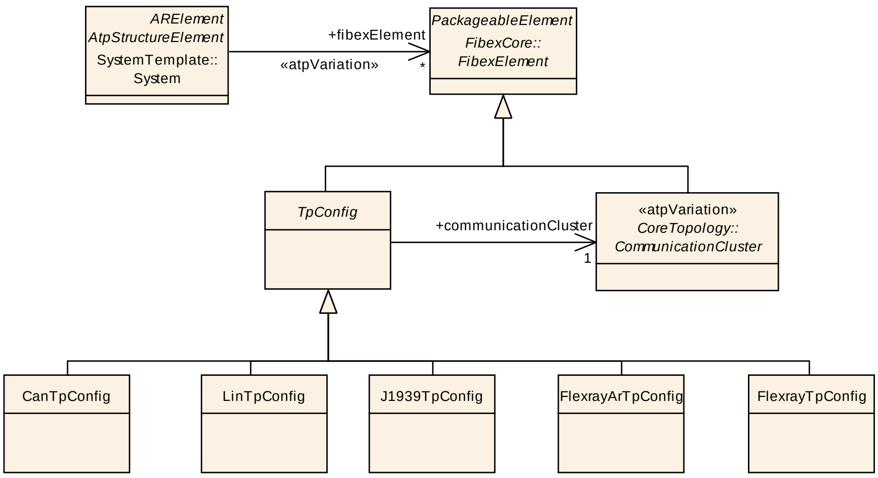

Figure 5.23: Transport Layer Overview

Examples in chapter 5.15.6 and chapter 5.15.7 illustrate the usage of the TP model.

In a normal case the PDU-routing is only supported for IPdus. In case of a gateway every incoming NPdus needs to be:

- forwarded to corresponding inbound TP module and transformed into an IPdu
- the IPdu needs to be forwarded to the PduR
- the PduR routes the IPdu to the outgoing TP module
- the outbound TP module transforms the IPdu into a NPdu which is then sent on the target bus.

Especially the transformations in the TP modules take a significant amount of time and resources. The behavior can be optimized when the source and the target network are of the same kind (e.g. Can-to-Can routing). In this case the inbound NPdu can be directly forwarded to the PduR and then sent on the outbound bus without any (resource consuming) TP module involvement. To support such an low level TP routing in the System Template the `NPdu` element is a specialization of the `IPdu` element. This allows the PDU-routing of `NPdus`.

Class	**TpConfig (abstract)**			
Package	M2::AUTOSARTemplates::SystemTemplate::TransportProtocols			
Note	Contains all configuration elements for AUTOSAR TP.			
Base	ARObject,CollectableElement,FibexElement,Identifiable,Multilanguage Referrable,PackageableElement,Referrable			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
communicationCluster	Communication Cluster	1	ref	A TpConfig is existing always in the context of exactly one CommunicationCluster.

Table 5.101: TpConfig

Class	**NPdu**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::FibexCore::CoreCommunication			
Note	This is a Pdu of the Transport Layer. The main purpose of the TP Layer is to segment and reassemble IPdus. In case of a Pdu Gateway when the source and the target network are of the same kind (e.g. Can-to-Can routing) it is possible to optimize the routing. The incoming NPdu can be directly forwarded to the PduR and then be sent on the outbound bus without any (resource consuming) TP module involvement. To support this use case the NPdu is located under the IPdu. But in the AUTOSAR Layered Architecture the NPdu is not a specialization of an IPdu. **Tags:** atp.recommendedPackage=Pdus			
Base	ARObject,CollectableElement,FibexElement,IPdu,Identifiable,Multilanguage Referrable,PackageableElement,Pdu,Referrable			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
–	–	–	–	–

Table 5.102: NPdu

Class	**TpAddress**			
Package	M2::AUTOSARTemplates::SystemTemplate::TransportProtocols			
Note	An ECUs TP address on the referenced channel. This represents the diagnostic Address.			
Base	ARObject,Identifiable,MultilanguageReferrable,Referrable			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
tpAddress	Integer	1	attr	An ECUs TP address on the referenced channel. This represents the diagnostic Address.

Table 5.103: TpAddress

[constr_3025] Usage of NPdus in TpConnections ⌈ In case several TpConnections use the same Frame ID for their communication needs there shall exist only one `NPdu` element per Frame Id. This constraint applies for all supported AUTOSAR transport protocols (CanTp, LinTp, FrTp and J1939Tp). ⌋

Note: Depending on the capabilities of the Basic Software implementations of Tp and Interface the ECU Configuration of the respective BSW Modules may utilize more communication elements (NPdus).

Example for an allowed System Template description where the same FrameId is used by two different TpConnections:

```
TpConnection1 --(dataPdu)--> NPdu1 ----> FrameId1
TpConnection1 --(flowControl)--> NPdu2 ----> FrameId2
TpConnection2 --(dataPdu)--> NPdu2 ----> FrameId2
TpConnection2 --(flowControl)--> NPdu1 ----> FrameId1
```

The following Ecu configuration with additional NPdus can still be derived from the above system description:

```
TpConnection1 --(dataPdu)--> NPdu1 ----> FrameId1
TpConnection1 --(flowControl)--> NPdu2 ----> FrameId2
TpConnection2 --(dataPdu)--> NPdu3 ----> FrameId2
TpConnection2 --(flowControl)--> NPdu4 ----> FrameId1
```


— AUTOSAR CONFIDENTIAL —

5.15.1 FlexRay ISO Transport Layer

The FlexRay ISO 10681-2 Transport Layer supports multiple sessions, i.e. multiple segmented transfers can be handled at the same time. Thus, multiple `FlexRayTpConnections` can be defined on the same ECU. Each `FlexRayTpConnection` is controlled by configuration parameters defined in `FlexRayTpConnectionControl`. The same `FlexRayTpConnectionControl` can be reused for an arbitrary number of `FlexRayTpConnections`.

A `FlexRayTpConnection` defines the way of communication between a sender and a receiver and uses a `FlexRayTpPduPool` of `NPdus` to transmit data to the FlexRay Interface. Each `FlexRayTpConnection` needs to specify one `txPduPool` with at least one `NPdu` as transmit PDU; however, in order to achieve a higher band width a `txPduPool` may contain more than one transmit `NPdu`.

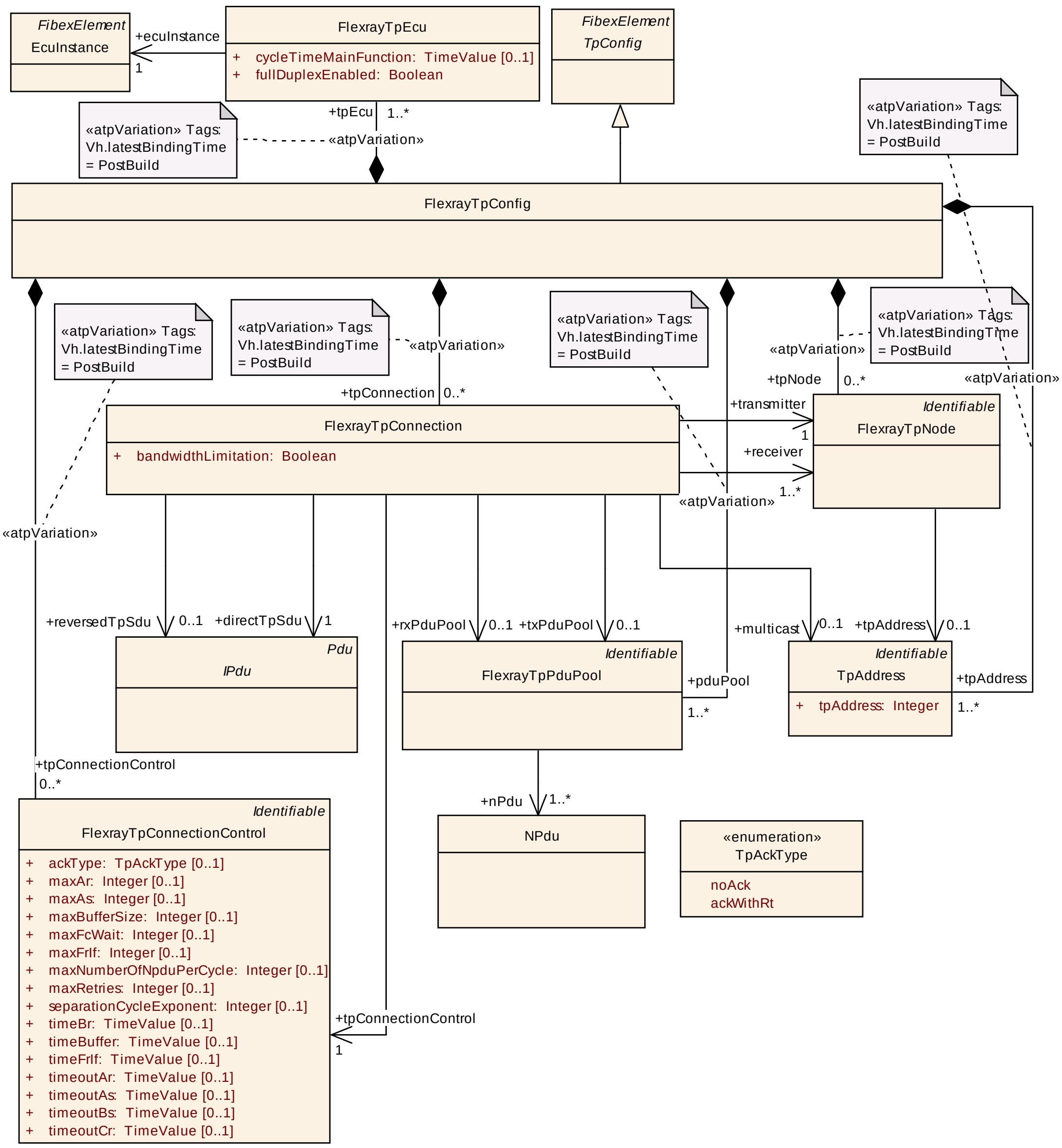

Figure 5.24: FlexRay ISO Transport Layer Configuration (TransportProtocols: FlexRayIsoTransportProtocol)

FlexRayTpConnections are specifically used for communication between one source and one or several target devices. These communication partners are specified using the source and target associations to FlexrayTpNodes, providing the diagnostic tpAddress and the connection to the topology. In case of several receivers a multicast tpAddress shall be used.

The actual payload to be transported by the FlexRayTpConnection is specified by using either one or two references to IPdus, depending on whether the connection shall be used unidirectional (one reference) or bidirectional (two references).

Class	**FlexrayTpConfig**			
Package	M2::AUTOSARTemplates::SystemTemplate::TransportProtocols			
Note	This element defines exactly one FlexRay ISO TP Configuration. One FlexRayTpConfig element shall be created for each FlexRay Network in the System that uses FlexRay Iso Tp. **Tags:** atp.recommendedPackage=TpConfigs			
Base	ARObject,CollectableElement,FibexElement,Identifiable,Multilanguage Referrable,PackageableElement,Referrable,TpConfig			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
pduPool	FlexrayTpPduPool	1..*	aggr	Configuration of FlexRay TP Pdu Pools. **Stereotypes:** atpVariation **Tags:** Vh.latestBindingTime=PostBuild
tpAddress	TpAddress	1..*	aggr	Collection of TpAddresses. atpVariation: Derived, because EcuInstance can vary. **Stereotypes:** atpVariation **Tags:** Vh.latestBindingTime=PostBuild
tpConnection	FlexrayTpConnection	*	aggr	Configuration of FlexRay TP Connections. atpVariation: Derived, because TpNode can vary. **Stereotypes:** atpVariation **Tags:** Vh.latestBindingTime=PostBuild
tpConnectionControl	FlexrayTpConnectionControl	*	aggr	Configuration of FlexRay TP Connection Controls. **Stereotypes:** atpVariation **Tags:** Vh.latestBindingTime=PostBuild
tpEcu	FlexrayTpEcu	1..*	aggr	Collection of TP Ecus atpVariation: Derived, because EcuInstance can vary. **Stereotypes:** atpVariation **Tags:** Vh.latestBindingTime=PostBuild

Attribute	Datatype	Mul.	Kind	Note
tpNode	FlexrayTpNode	*	aggr	Senders and receivers of FlexRay TP messages. atpVariation: Derived, because EcuInstance can vary. **Stereotypes:** atpVariation **Tags:** Vh.latestBindingTime=PostBuild

Table 5.104: FlexrayTpConfig

Class	**FlexrayTpConnectionControl**			
Package	M2::AUTOSARTemplates::SystemTemplate::TransportProtocols			
Note	Configuration parameters to control a FlexRay TP connection.			
Base	ARObject,Identifiable,MultilanguageReferrable,Referrable			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**
ackType	TpAckType	0..1	attr	This parameter defines the type of acknowledgement which is used for the specific channel.
maxAr	Integer	0..1	attr	This parameter defines the maximum number of trying to send a frame when a TIMEOUT AR occurs (depending on whether retry is configured).
maxAs	Integer	0..1	attr	This parameter defines the maximum number of trying to send a frame when a TIMEOUT AS occurs (depending on whether retry is configured)
maxBuffer Size	Integer	0..1	attr	This parameter is only relevant when having retry activated. It limits the maximal buffer size the FrTp can choose in order to limit the amount of Tx buffer that will be requested at the sender side in a segmented transfer.
maxFcWait	Integer	0..1	attr	This attribute defines the maximum number of FlowControl N-PDUs with FlowState "WAIT".
maxFrIf	Integer	0..1	attr	This parameter defines the maximum number of trying to send a frame when the FrIf returns an error
maxNumb erOfNpduP erCycle	Integer	0..1	attr	This parameter limits the number of N-Pdus the sender is allowed to transmit within a FlexRay cycle.
maxRetrie s	Integer	0..1	attr	This parameter defines the maximum number of retries (if retry is configured for the particular channel).
separation CycleExpo nent	Integer	0..1	attr	Exponent to calculate the minimum number of "Separation Cycles" the sender has to wait for the next transmission of an FrTp N-Pdu.
timeBr	TimeValue	0..1	attr	Time (in seconds) until transmission of the next FlowControl N-PDU.

Attribute	Datatype	Mul.	Kind	Note
timeBuffer	TimeValue	0..1	attr	This parameter defines the time of waiting for the next try to get a Tx or Rx buffer. This parameter is equivalent to the temporal distance between two FC.WT N-Pdus in case the buffer request returns busy. Specified in seconds.
timeFrIf	TimeValue	0..1	attr	This parameter defines the time of waiting for the next try to send. Specified in seconds.
timeoutAr	TimeValue	0..1	attr	This parameter states the timeout between the PDU transmit request of the Transport Layer to the FlexRay Interface and the corresponding confirmation of the FlexRay Interface on the receiver side (for FC or AF). Specified in seconds.
timeoutAs	TimeValue	0..1	attr	This attribute states the timeout between the PDU transmit request for the first PDU of the group used in the current connection of the Transport Layer to the FlexRay Interface and the corresponding confirmation of the FlexRay Interface (when having sent the last PDU of the group used in this connection) on the sender side (SF-x, FF-x, CF or FC (in case of Transmit Cancellation)). Specified in seconds.
timeoutBs	TimeValue	0..1	attr	This parameter defines the timeout in seconds for waiting for an FC or AF on the sender side in a 1:1 connection.
timeoutCr	TimeValue	0..1	attr	This parameter defines the timeout value in seconds for waiting for a CF or FF-x (in case of retry) after receiving the last CF or after sending an FC or AF on the receiver side. Specified in seconds.

Table 5.105: FlexrayTpConnectionControl

Class	**FlexrayTpConnection**			
Package	M2::AUTOSARTemplates::SystemTemplate::TransportProtocols			
Note	A connection identifies the sender and the receiver of this particular communication. The FlexRayTp module routes a Pdu through this connection. In a System Description the references to the PduPools are mandatory. In an ECU Extract these references can be optional: On unicast connections these references are always mandatory. On multicast the txPduPool is mandatory on the sender side. The rxPduPool is mandatory on the receiver side. On Gateway ECUs both references are mandatory.			
Base	ARObject			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**
bandwidth Limitation	Boolean	1	attr	Specifies whether the connection requires a bandwidth limitation or not.

Attribute	Datatype	Mul.	Kind	Note
directTpSdu	IPdu	1	ref	Reference to the IPdu that is segmented by the Transport Protocol. To support the low-level routing of NPdu's the NPdu is a specialization of an IPdu. More details can be found in the NPdu class description. Nevertheless the FlexrayTpConnection must not reference a NPdu with this tpSdu reference.
multicast	TpAddress	0..1	ref	TP address for 1:n connections.
receiver	FlexrayTpNode	1..*	ref	The target of the TP connection.
reversedTpSdu	IPdu	0..1	ref	Reference to the IPdu that is segmented by the Transport Protocol. If support of both sending and receiving is used, this association references the IPdu used for the additional second direction. To support the low-level routing of NPdu's the NPdu is a specialization of an IPdu. More details can be found in the NPdu class description. Nevertheless the FlexrayTpConnection must not reference a NPdu with this tpSdu reference.
rxPduPool	FlexrayTpPduPool	0..1	ref	A connection has a reference to a set of NPdus (FrTpRxPduPool) which are defined for receiving data via this particular connection. The following constraint is valid only for the System Extract/ECU Extract: In case this connection is applied to the transmitter the rxPduPool holds the actually received NPdus. In case this connection is applied to the receiver the rxPduPool holds the actually sent NPdus.
tpConnectionControl	FlexrayTpConnectionControl	1	ref	Reference to the connection control.
transmitter	FlexrayTpNode	1	ref	The source of the TP connection.
txPduPool	FlexrayTpPduPool	0..1	ref	A connection has a reference to a set of NPdus (FrTpTxPduPool) which are defined for sending data via this particular connection. The following constraint is valid only for the System Extract/ECU Extract: In case this connection is applied to the transmitter the txPduPool holds the actually sent NPdus. In case this connection is applied to the receiver the txPduPool holds the actually received NPdus.

Table 5.106: FlexrayTpConnection

The FlexrayTpConnection refers to the FlexrayTpPduPool in two roles: "rxPduPool" and "txPduPool". In the System/ECU Extract the information that is kept in the PduPools depends on the role of the regarded ECU: If the ECU is the transmitter then the rxPduPool holds the received NPdus and the txPduPool holds the sent NPdus. If the ECU is the receiver then the rxPduPool holds the sent NPdus and the txPduPool holds the received NPdus. The following example shows how this differentiation may be used:

System Description: SENDER = A
RECEIVER = B
TxPool = PDU_1
RxPool = PDU_2

ECU Extract of A:
SENDER = A
TxPool = PDU_1
RxPool = PDU_2

Since on receiver side the PDU_1 is received and PDU_2 is sent (from a local point of view) the export might look like this:
ECU Extract of B: RECEIVER = B
RxPool = PDU_1
TxPool = PDU_2

Class	**FlexrayTpPduPool**			
Package	M2::AUTOSARTemplates::SystemTemplate::TransportProtocols			
Note	FlexrayTpPduPool is a set of N-PDUs which are defined for FrTp sending or receiving purpose.			
Base	ARObject,Identifiable,MultilanguageReferrable,Referrable			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
nPdu	NPdu	1..*	ref	Reference to NPdus that are part of the PduPool.

Table 5.107: FlexrayTpPduPool

Class	**FlexrayTpNode**			
Package	M2::AUTOSARTemplates::SystemTemplate::TransportProtocols			
Note	TP Node (Sender or Receiver) provides the TP Address and the connection to the Topology description.			
Base	ARObject,Identifiable,MultilanguageReferrable,Referrable			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
connector	Communication Connector	*	ref	Asscociation to one or more physical connectors (max number of connectors for FlexRay: 2). In a System Description this reference is mandatory. In an ECU Extract this reference is optional (references to ECUs that are not part of the ECU Extract shall be avoided).
tpAddress	TpAddress	0..1	ref	Reference to the TP Address that is used by the TpNode. This reference is optional in case that the multicast TP Address is used (reference from TpConnection).

Table 5.108: FlexrayTpNode

Class	**FlexrayTpEcu**			
Package	M2::AUTOSARTemplates::SystemTemplate::TransportProtocols			
Note	ECU specific TP configuration parameters. Each TpEcu element has a reference to exactly one ECUInstance in the topology.			
Base	ARObject			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**
cycleTimeMainFunction	TimeValue	0..1	attr	The period between successive calls to the Main Function of the AUTOSAR TP. Specified in seconds.
ecuInstance	EcuInstance	1	ref	Connection to the ECUInstance in the Topology
fullDuplexEnabled	Boolean	1	attr	The full duplex mechanisms is enabled if this attribute is set to true. Otherwise half duplex is enabled.

Table 5.109: FlexrayTpEcu

5.15.2 FlexRay AUTOSAR Transport Layer

This section describes a Non-ISO FlexRay TP protocol that is supported by AUTOSAR in addition to the FlexRay ISO 10681-2 TP (see section 5.15.1). The Non-ISO FlexRay Transport Layer supports multiple sessions, i.e. multiple segmented transfers can be handled at the same time. As each of these sessions requires individual state machines and thus additional resources, the same session (in FlexRay TP called FrArTpChannel) can be reused for an arbitrary number of FlexrayArTpConnections.

A FlexRayArTpChannel provides a pool of NPdus which may are being used by the channels FlexRayArTpConnections: Each FlexRayArTpConnection needs to specify at least one NPdu as transmit Pdu; however, in order to achieve a higher band width the same connection may use more than one transmit NPdu.

As there is no concurrent transfer of connections within one channel, a flow control NPdu can be specified globally for the FlexRayArTpChannel. In this case, all FlexRayArTpConnections being realized by this channel use the same NPdu for Flow Control. However, each FlexRayArTpConnection may also define its own flow control NPdu. FlexRayTpConnections are specifically used for communication between one source and one target device. These communication partners are specified using the source and target associations to FlexrayArTpNodes, providing the diagnostic TpAddress and the connection to the topology description. The actual payload to be transported by the FlexRayArTpConnection is specified by using either one or two references to IPdus, depending on whether the connection shall be used unidirectional (one reference) or bidirectional (two references).

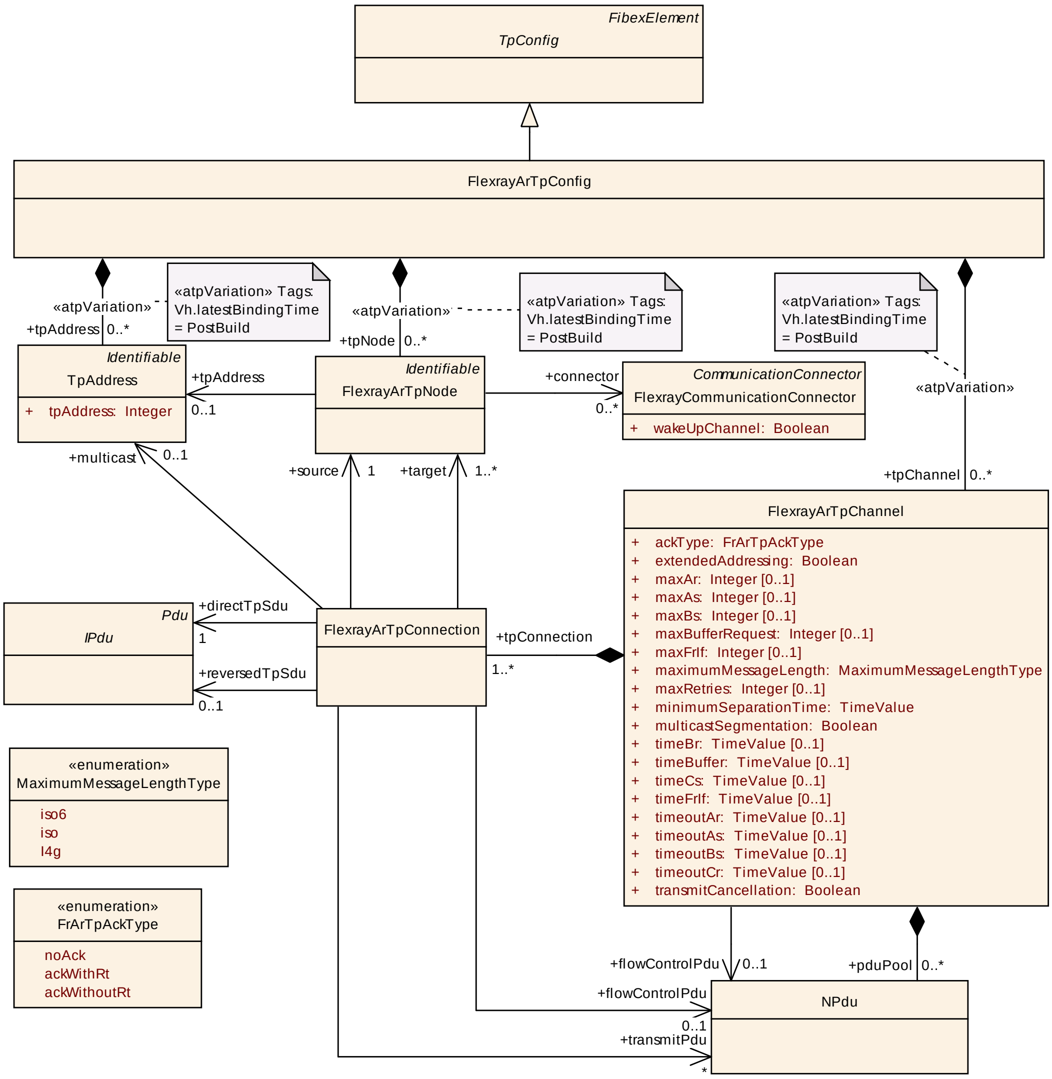

Figure 5.25: FlexRay Autosar Transport Layer Configuration (TransportProtocols: FlexRayAutosarTransportProtocol)

Class	**FlexrayArTpChannel**			
Package	M2::AUTOSARTemplates::SystemTemplate::TransportProtocols			
Note	A channel is a group of connections sharing several properties. The FlexRay AutosarTransport Layer supports several channels. These channels can work concurrently, thus each of them requires its own state machine and management data structures and its own PDU-IDs.			
Base	ARObject			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
ackType	FrArTpAckType	1	attr	Type of Acknowledgement.
extendedAddressing	Boolean	1	attr	Adressing Type of this connection: true: Two Bytes false: One Byte

Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
flowControlPdu	NPdu	0..1	ref	Reference to the Flow Control NPdu. The Flow Control network protocol data unit (FC N_PDU) is identified by the Flow Control protocol control information (FC N_PCI). The Flow Control network protocol data unit (FC N_PDU) instructs a sending network entity to start, stop or resume transmission of CF N_PDUs.
maxAr	Integer	0..1	attr	This attribute defines the maximum number of trying to send a frame when a TIMEOUT AR occurs (depending on whether retry is configured).
maxAs	Integer	0..1	attr	This attribute defines the maximum number of trying to send a frame when a TIMEOUT AS occurs (depending on whether retry is configured).
maxBs	Integer	0..1	attr	This attribute limits the maximal block size the FrTp can choose in order to limit the amount of Tx buffer that will be requested at the sender side in a segmented transfer. range when retry is activated: 1-16 range when retry is not activated: 0-255
maxBufferRequest	Integer	0..1	attr	This attribute defines the maximum number of trying to get a buffer (Transmit / Receive), depending of the return value of PduR_FrTpProvideTxBuffer / PduR_FrTpProvideRxBuffer and on whether retry is configured.
maxFrIf	Integer	0..1	attr	This attribute defines the maximum number of trying to send a frame when the FrIf returns an error.
maxRetries	Integer	0..1	attr	This attribute defines the maximum number of retries (if retry is configured for the particular channel).
maximumMessageLength	MaximumMessageLengthType	1	attr	This specifies the maximum message length for the particular channel.
minimumSeparationTime	TimeValue	1	attr	This attribute defines the minimum amount of time (separation Time) between two succeeding CFs. Specified in seconds.
multicastSegmentation	Boolean	1	attr	This attribute defines whether segmentation within a 1:n connection is allowed or not.
pduPool	NPdu	*	aggr	A FlexRayTpChannel contains a pool of NPdus.
timeBr	TimeValue	0..1	attr	This attribute defines the time in seconds between receiving the last CF of a block or an FF-x (or SF-x) and sending out an FC or AF.
timeBuffer	TimeValue	0..1	attr	This attribute defines the time in seconds of waiting for the next try (if retry is activated) to get a Tx or Rx buffer.

Attribute	Datatype	Mul.	Kind	Note
timeCs	TimeValue	0..1	attr	This attribute defines the time in seconds between the sending of two consecutive frames or between a consecutive frame and a flow control (for Transmit Cancellation) or between reception of an flow control or Acknowledgement Frame and sending of the next consecutive frame or a flow control (for Transmit Cancellation).
timeFrIf	TimeValue	0..1	attr	This attribute defines the time in seconds of waiting for the next try (if retry is activated) to send via FrIf_Transmit. Specified in seconds.
timeoutAr	TimeValue	0..1	attr	This attribute states the timeout in seconds between the PDU transmit request of the Transport Layer to the FlexRay Interface and the corresponding confirmation of the FlexRay Interface on the receiver side (for FC or AF).
timeoutAs	TimeValue	0..1	attr	This attribute states the timeout in seconds between the PDU transmit request for the first PDU of the group used in the current connection of the Transport Layer to the FlexRay Interface and the corresponding confirmation of the FlexRay Interface (when having sent the last PDU of the group used in this connection) on the sender side (SF-x, FF-x, CF).
timeoutBs	TimeValue	0..1	attr	This attribute defines the timeout in seconds for waiting for an FC or AF on the sender side in a 1:1 connection.
timeoutCr	TimeValue	0..1	attr	This attribute defines the timeout value in seconds for waiting for a CF or FF-x (in case of retry) after receiving the last CF or after sending an FC or AF on the receiver side.
tpConnection	FlexrayArTpConnection	1..*	aggr	Group of connections that can be used in this channel.
transmitCancellation	Boolean	1	attr	This attribute states whether Transmit Cancellation is supported on this channel.

Table 5.110: FlexrayArTpChannel

Class	FlexrayArTpNode			
Package	M2::AUTOSARTemplates::SystemTemplate::TransportProtocols			
Note	TP Node (Sender or Receiver) provides the TP Address and the connection to the Topology description.			
Base	ARObject,Identifiable,MultilanguageReferrable,Referrable			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**
connector	FlexrayCommunicationConnector	*	ref	Association to one or more physical connectors (max number of connectors for FlexRay: 2). In a System Description this reference is mandatory. In an ECU Extract this reference is optional (references to ECUs that are not part of the ECU Extract shall be avoided).

Attribute	Datatype	Mul.	Kind	Note
tpAddress	TpAddress	0..1	ref	Reference to the TP Address that is used by the TpNode. This reference is optional in case that the multicast TP Address is used (reference from TpConnection).

Table 5.111: FlexrayArTpNode

Class	FlexrayArTpConnection			
Package	M2::AUTOSARTemplates::SystemTemplate::TransportProtocols			
Note	A connection within a channel identifies the sender and the receiver of this particular communication. The FlexRay Autosar Tp module routes a Pdu through this connection.			
Base	ARObject			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**
directTpSdu	IPdu	1	ref	Reference to the IPdu that is segmented by the Transport Protocol. The source address of the transmitted NPdu is determined by the configured source CommunicationConnector. The target address of the transmitted NPdu is determined by the configured target CommunicationConnector. To support the low-level routing of NPdu's the NPdu is a specialization of an IPdu. More details can be found in the NPdu class description. Nevertheless the FlexRayTpConnection shall not reference a NPdu with this tpSdu reference.
flowControlPdu	NPdu	0..1	ref	Reference to the Flow Control NPdu. The Flow Control network protocol data unit (FC N_PDU) is identified by the Flow Control protocol control information (FC N_PCI). The Flow Control network protocol data unit (FC N_PDU) instructs a sending network entity to start, stop or resume transmission of CF N_PDUs. The Flow Control network protocol data unit shall be sent by the receiving network layer entity to the sending network layer entity, when ready to receive more data, after correct reception of: a) First Frame network protocol data unit (FF N_PDU) b) the last Consecutive Frame network protocol data unit (CF N_PDU) of a block of Consecutive Frames (CF N_PDU) if further Consecutive Frame network protocol data unit (CF N_PDU) need(s) to be sent.
multicast	TpAddress	0..1	ref	TP address for 1:n connections.

Attribute	Datatype	Mul.	Kind	Note
reversedTpSdu	IPdu	0..1	ref	Reference to the IPdu that is segmented by the Transport Protocol. If support of both sending and receiving is used, this association references the IPdu used for the additional second direction. The source address of the transmitted NPdu is determined by the configured target CommunicationConnector. The target address of the transmitted NPdu is determined by the configured source CommunicationConnector. To support the low-level routing of NPdu's the NPdu is a specialization of an IPdu. More details can be found in the NPdu class description. Nevertheless the FlexRayTpConnection shall not reference a NPdu with this tpSdu reference.
source	FlexrayArTpNode	1	ref	The source of the TP connection.
target	FlexrayArTpNode	1..*	ref	The target of the TP connection.
transmitPdu	NPdu	*	ref	Reference to an NPdu (Single Frame, First Frame or Consecutive Frame). The Single Frame network protocol data unit (SF N_PDU) shall be sent out by the sending network entity and can be received by one or multiple receiving network entities. The Single Frame (SF N_PDU) shall be sent out to transfer a service data unit that can be transferred via a single service request to the data link layer. This network protocol data unit shall be sent to transfer unsegmented messages. The First Frame network protocol data unit (FF N_PDU) identifies the first network protocol data unit (N_PDU) of a segmented message transmitted by a network sending entity and received by a receiving network entity. The Consecutive Frame network protocol data unit (CF N_PDU) transfers segments (N_Data) of the service data unit message data (<MessageData>). All network protocol data units (N_PDUs) transmitted by the sending entity after the First Frame network protocol data unit (FF N_PDU) shall be encoded as Consecutive Frames network protocol data units (CF N_PDUs).

Table 5.112: FlexrayArTpConnection

5.15.3 CAN Transport Layer

The CAN Transport Layer supports multiple sessions by means of so called CanTpChannels: Each CanTpChannel uses its own resources, such as internal buffer, timer, state machine and thus can operate independently and simultaneously to other CanTpChannels. The same session can be reused for an arbitrary number of CanTpConnections.

Each CanTpConnection uses its own pair of NPdus: One NPdu, the dataPdu is mandatory for each CanTpConnection, the flowControlPdu is optional depending whether only Single Frames are transferred over the connection.

A CanTpConnection is specifically used for communication between one source and one or several target devices. These communication partners are specified using the source and target associations to CanTpNode, providing the diagnostic tpAddress and the connection to the topology. In case of several receivers a multicast tpAddress shall be used.

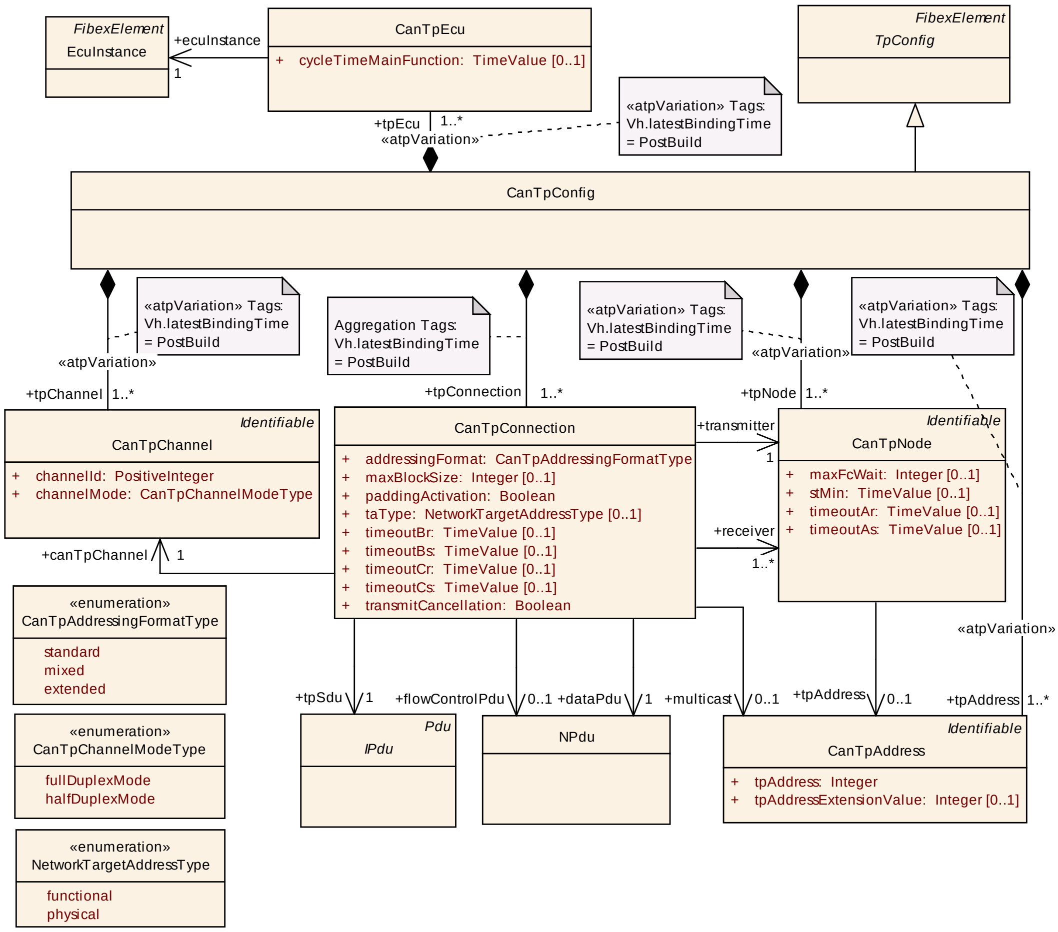

Figure 5.26: CAN Transport Layer Configuration (TransportProtocols: CanTransportProtocol)

The actual payload to be transported by the CanTpConnection is specified by the reference tpSdu to IPdu.

Class	**CanTpConfig**			
Package	M2::AUTOSARTemplates::SystemTemplate::TransportProtocols			
Note	This element defines exactly one CAN TP Configuration. One CanTpConfig element shall be created for each CAN Network in the System. **Tags:** atp.recommendedPackage=TpConfigs			
Base	ARObject,CollectableElement,FibexElement,Identifiable,Multilanguage Referrable,PackageableElement,Referrable,TpConfig			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**
tpAddress	CanTpAddress	1..*	aggr	Collection of TP Adresses. atpVariation: Derived, because EcuInstance can vary. **Stereotypes:** atpVariation **Tags:** Vh.latestBindingTime=PostBuild
tpChannel	CanTpChannel	1..*	aggr	Configuration of CAN TP channels. **Stereotypes:** atpVariation **Tags:** Vh.latestBindingTime=PostBuild
tpConnection	CanTpConnection	1..*	aggr	Senders and receivers of CAN TP messages. atpVariation: Derived, because TpNode can vary. **Tags:** Vh.latestBindingTime=PostBuild
tpEcu	CanTpEcu	1..*	aggr	Collection of TP Ecus atpVariation: Derived, because EcuInstance can vary. **Stereotypes:** atpVariation **Tags:** Vh.latestBindingTime=PostBuild
tpNode	CanTpNode	1..*	aggr	Senders and receivers of Can TP messages. atpVariation: Derived, because EcuInstance can vary. **Stereotypes:** atpVariation **Tags:** Vh.latestBindingTime=PostBuild

Table 5.113: CanTpConfig

Class	**CanTpChannel**			
Package	M2::AUTOSARTemplates::SystemTemplate::TransportProtocols			
Note	Configuration parameters of the CanTp channel.			
Base	ARObject,Identifiable,MultilanguageReferrable,Referrable			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**
channelId	PositiveInteger	1	attr	The id of the channel. The value shall be unique for each channel.
channelMode	CanTpChannelModeType	1	attr	The CAN Transport Layer supports half and full duplex channel modes.

Table 5.114: CanTpChannel

Enumeration	**CanTpChannelModeType**
Package	M2::AUTOSARTemplates::SystemTemplate::TransportProtocols
Note	The CAN Transport Layer supports half and full duplex channel modes.
Literal	**Description**
fullDuplexMode	full duplex channel mode
halfDuplexMode	half duplex channel mode

Table 5.115: CanTpChannelModeType

Class	**CanTpConnection**			
Package	M2::AUTOSARTemplates::SystemTemplate::TransportProtocols			
Note	A connection identifies the sender and the receiver of this particular communication. The CanTp module routes a Pdu through this connection. atpVariation: Derived, because TpNode can vary.			
Base	ARObject			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**
addressingFormat	CanTpAddressingFormatType	1	attr	Declares which communication addressing mode is supported.
canTpChannel	CanTpChannel	1	ref	Reference to the CanTpChannel on which this CanTpConnection is realized.
dataPdu	NPdu	1	ref	Reference to an Data NPdu.
flowControlPdu	NPdu	0..1	ref	Reference to the Flow Control NPdu.
maxBlockSize	Integer	0..1	attr	The maximum number of N-PDUs the CanTp receiver allows the sender to send, before waiting for an authorization to continue transmission of the following N-PDUs. For further details on this parameter value see ISO 15765-2 specification. Note: For reasons of buffer length, the CAN Transport Layer can adapt the BS value within the limit of this maximum BS
multicast	CanTpAddress	0..1	ref	TP address for 1:n connections.

Attribute	Datatype	Mul.	Kind	Note
paddingActivation	Boolean	1	attr	This specifies wheter or not Sfs, FCs and the last CF shall be padded to 8 bytes length in case it contains less payload. true: The N-PDU received uses padding for SF, FC and the last CF. (N-PDU length is always 8 bytes) false: The N-PDU received does not use padding for SF, CF and the last CF. (N-PDU length is dynamic)
receiver	CanTpNode	1..*	ref	The target of the TP connection.
taType	NetworkTargetAddressType	0..1	attr	Network Target Address type.
timeoutBr	TimeValue	0..1	attr	Value in seconds of the performance requirement for (N_Br + N_Ar). N_Br is the elapsed time between the receiving indication of a FF or CF or the transmit confirmation of a FC, until the transmit request of the next FC.
timeoutBs	TimeValue	0..1	attr	This parameter defines the timout for waiting for an FC or AF on the sender side in an 1:1 connection. Specified in seconds.
timeoutCr	TimeValue	0..1	attr	This parameter defines the timeout value for waiting for a CF or FF-x (in case of retry) after receiving the last CF or after sending an FC or AF on the receiver side. Specified in seconds.
timeoutCs	TimeValue	0..1	attr	The attribute timeoutCs represents the time (in seconds) which elapses between the transmit request of a CF N-PDU until the transmit request of the next CF N-PDU.
tpSdu	IPdu	1	ref	Reference to an IPdu that is segmented by the Transport Protocol. To support the low-level routing of NPdu's the NPdu is a specialization of an IPdu. More details can be found in the NPdu class description. Nevertheless the CanTpConnection must not reference a NPdu with this tpSdu reference.
transmitCancellation	Boolean	1	attr	With this switch Transmit Cancellation can be turned on or off for this channel.
transmitter	CanTpNode	1	ref	The source of the TP connection.

Table 5.116: CanTpConnection

Enumeration	**CanTpAddressingFormatType**
Package	M2::AUTOSARTemplates::SystemTemplate::TransportProtocols
Note	Declares which communication addressing mode is supported.
Literal	**Description**
extended	To use extended addressing format.
mixed	To use mixed addressing format.
standard	To use normal addressing format.

Table 5.117: CanTpAddressingFormatType

Class	**CanTpAddress**			
Package	M2::AUTOSARTemplates::SystemTemplate::TransportProtocols			
Note	An ECUs TP address on the referenced channel. This represents the diagnostic Address.			
Base	ARObject,Identifiable,MultilanguageReferrable,Referrable			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**
tpAddress	Integer	1	attr	An ECUs TP address on the referenced channel. This represents the diagnostic Address.
tpAddress Extension Value	Integer	0..1	attr	If the mixed addressing format is used, this parameter contains the transport protocol address extension value.

Table 5.118: CanTpAddress

Class	**CanTpEcu**			
Package	M2::AUTOSARTemplates::SystemTemplate::TransportProtocols			
Note	ECU specific TP configuration parameters. Each TpEcu element has a reference to exactly one ECUInstance in the topology.			
Base	ARObject			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**
cycleTime MainFuncti on	TimeValue	0..1	attr	The period between successive calls to the Main Function of the AUTOSAR TP. Specified in seconds.
ecuInstanc e	EcuInstance	1	ref	Connection to the ECUInstance in the Topology

Table 5.119: CanTpEcu

Class	**CanTpNode**			
Package	M2::AUTOSARTemplates::SystemTemplate::TransportProtocols			
Note	TP Node (Sender or Receiver) provides the TP Address and the connection to the Topology description.			
Base	ARObject,Identifiable,MultilanguageReferrable,Referrable			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**

— AUTOSAR CONFIDENTIAL —

Attribute	*Datatype*	*Mul.*	*Kind*	*Note*
connector	Communication Connector	0..1	ref	Asscociation to a CommunicationConnector in the topology description. In a System Description this reference is mandatory. In an ECU Extract this reference is optional (references to ECUs that are not part of the ECU Extract shall be avoided).
maxFcWait	Integer	0..1	attr	This attribute defines the maximum number of flow control PDUs that can be consecutively be transmitted by a receiver.
stMin	TimeValue	0..1	attr	Sets the duration of the minimum time the CanTp sender shall wait between the transmissions of two CF N-PDUs.
timeoutAr	TimeValue	0..1	attr	This attribute states the timeout between the PDU transmit request of the Transport Layer to the Can Interface and the corresponding confirmation of the Can Interface on the receiver side (for FC or AF). Specified in seconds.
timeoutAs	TimeValue	0..1	attr	This attribute states the timeout between the PDU transmit request for the first PDU of the group used in the current connection of the Transport Layer to the Can Interface and the corresponding confirmation of the Can Interface (when having sent the last PDU of the group used in this connection) on the sender side (SF-x, FF-x, CF or FC (in case of Transmit Cancellation)). Specified in seconds.
tpAddress	CanTpAddress	0..1	ref	Reference to the TP Address that is used by the TpNode. This reference is optional in case that the multicast TP Address is used (reference from TpConnection).

Table 5.120: CanTpNode

5.15.4 LIN Transport Layer

LinTpConnection is used for modeling communication resources required for using the LIN Transport Layer. Contrary to the FlexRay and CAN Transport Layers, LIN TP only supports one session per PhysicalChannel. Therefore it is a semantical constraint that maximal two LinTpConnections can be defined per LIN Cluster: one LinTpConnection to describe the transmission of data from master to slave, using the MasterRequest frame, and the other LinTpConnection to describe the transmission of data from slave to master, using the SlaveResponse frame.

LinTpConnection uses the dataPdu reference for specifying exactly one NPdu which is to be used for transmitting the data, and it optionally references a flowControl NPdu in order to handle Flow Control Frames if required.

One LinTpConnection is specifically used for communication between one source and one or several target devices. These communication partners are specified using the source and target associations to LinTpNode, providing the diagnostic tpAddress and the connection to the topology. In case of several receivers a multicast tpAddress shall be used.

The actual payload to be transported by the LinTpConnection is specified by the reference linTpNSdu to IPdu.

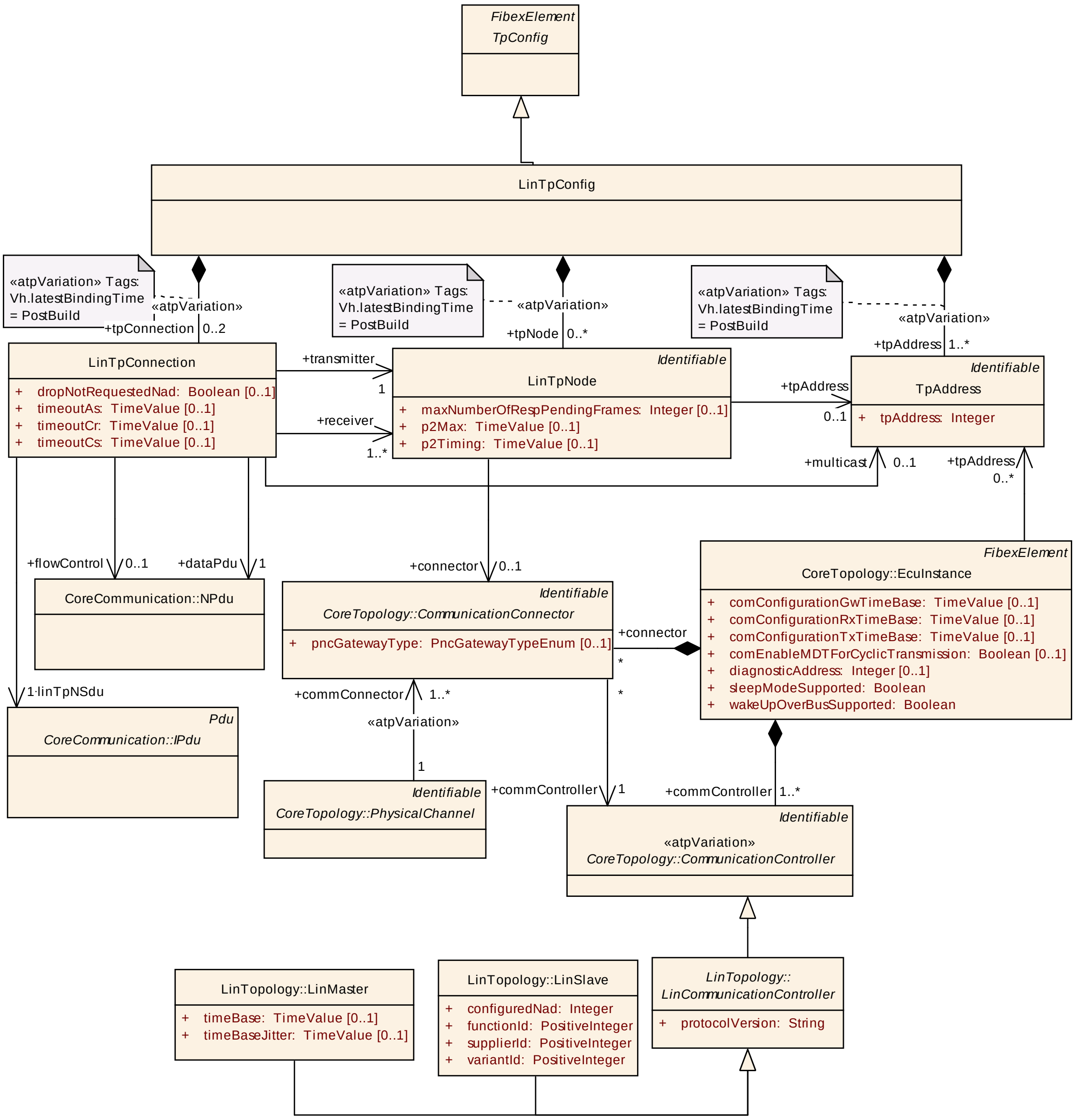

Figure 5.27: LIN Transport Layer Configuration

Class	**LinTpConfig**			
Package	M2::AUTOSARTemplates::SystemTemplate::TransportProtocols			
Note	This element defines exactly one Lin TP Configuration. One LinTpConfig element shall be created for each Lin Network in the System. **Tags:** atp.recommendedPackage=TpConfigs			
Base	ARObject,CollectableElement,FibexElement,Identifiable,MultilanguageReferrable,PackageableElement,Referrable,TpConfig			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**
tpAddress	TpAddress	1..*	aggr	Collection of TpAddresses. atpVariation: Derived, because EcuInstance can vary. **Stereotypes:** atpVariation **Tags:** Vh.latestBindingTime=PostBuild
tpConnection	LinTpConnection	0..2	aggr	Configuration of LIN TP channels. atpVariation: Derived, because TpNode can vary. **Stereotypes:** atpVariation **Tags:** Vh.latestBindingTime=PostBuild
tpNode	LinTpNode	*	aggr	Senders and receivers of LIN TP messages. atpVariation: Derived, because EcuInstance can vary. **Stereotypes:** atpVariation **Tags:** Vh.latestBindingTime=PostBuild

Table 5.121: LinTpConfig

Class	**LinTpNode**			
Package	M2::AUTOSARTemplates::SystemTemplate::TransportProtocols			
Note	TP Node (Sender or Receiver) provides the TP Address and the connection to the Topology description.			
Base	ARObject,Identifiable,MultilanguageReferrable,Referrable			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**
connector	CommunicationConnector	0..1	ref	Asscociation to a CommunicationConnector in the topology description. In a System Description this reference is mandatory. In an ECU Extract this reference is optional (references to ECUs that are not part of the ECU Extract shall be avoided).
maxNumberOfRespPendingFrames	Integer	0..1	attr	Configures the maximum number of allowed response pending frames.

Attribute	Datatype	Mul.	Kind	Note
p2Max	TimeValue	0..1	attr	After reception of a response pending frame the P2 timeout counter is reloaded with the timeout time P2max.
p2Timing	TimeValue	0..1	attr	P2 timeout observation parameter.
tpAddress	TpAddress	0..1	ref	Reference to the TP Address that is used by the TpNode. This reference is optional in case that the multicast TP Address is used (reference from TpConnection).

Table 5.122: LinTpNode

Class	LinTpConnection			
Package	M2::AUTOSARTemplates::SystemTemplate::TransportProtocols			
Note	A LinTP channel represents an internal path for the transmission or reception of a Pdu via LinTp and describes the the sender and the receiver of this particular communication. Two LinTpConnections can be specified: one LinTpConnection to describe the transmission of data from master to slave, using MasterRequestFrame and one LinTpConnection to describe the transmission of data from slave to master, using the SlaveResponseFrame.			
Base	ARObject			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**
dataPdu	NPdu	1	ref	Reference to an NPdu (Single Frame, First Frame or Consecutive Frame). The Single Frame network protocol data unit (SF N_PDU) shall be sent out by the sending network entity and can be received by one or multiple receiving network entities. The Single Frame (SF N_PDU) shall be sent out to transfer a service data unit that can be transferred via a single service request to the data link layer. This network protocol data unit shall be sent to transfer unsegmented messages. The First Frame network protocol data unit (FF N_PDU) identifies the first network protocol data unit (N_PDU) of a segmented message transmitted by a network sending entity and received by a receiving network entity. The Consecutive Frame network protocol data unit (CF N_PDU) transfers segments (N_Data) of the service data unit message data (<MessageData>). All network protocol data units (N_PDUs) transmitted by the sending entity after the First Frame network protocol data unit (FF N_PDU) shall be encoded as Consecutive Frames network protocol data units (CF N_PDUs).

Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
dropNotRequestedNad	Boolean	0..1	attr	Configures if TP Frames of not requested LIN-Slaves are dropped or not.
flowControl	NPdu	0..1	ref	Reference to the Flow Control NPdu. The Flow Control network protocol data unit (FC N_PDU) is identified by the Flow Control protocol control information (FC N_PCI). The Flow Control network protocol data unit (FC N_PDU) instructs a sending network entity to start, stop or resume transmission of CF N_PDUs. The Flow Control network protocol data unit shall be sent by the receiving network layer entity to the sending network layer entity, when ready to receive more data, after correct reception of: a) First Frame network protocol data unit (FF N_PDU) b) the last Consecutive Frame network protocol data unit (CF N_PDU) of a block of Consecutive Frames (CF N_PDU) if further Consecutive Frame network protocol data unit (CF N_PDU) need(s) to be sent.
linTpNSdu	IPdu	1	ref	Reference to the IPdu that is segmented by the Transport Protocol. To support the low-level routing of NPdu's the NPdu is a specialization of an IPdu. More details can be found in the NPdu class description. Nevertheless the LinTpConnection must not reference a NPdu with this linTpNSdu reference.
multicast	TpAddress	0..1	ref	TP address for 1:n connections.
receiver	LinTpNode	1..*	ref	The target of the TP connection.
timeoutAs	TimeValue	0..1	attr	Time for transmission of the LIN frame (any N-PDU) on the sender side. Specified in seconds.
timeoutCr	TimeValue	0..1	attr	This attribute defines the timeout value for waiting for a CF or FF-x (in case of retry) after receiving the last CF or after sending an FC or AF on the receiver side. Specified in seconds.
timeoutCs	TimeValue	0..1	attr	The attribute timeoutCs represents the time (in seconds) which elapses between the transmit request of a CF N-PDU until the transmit request of the next CF N-PDU.
transmitter	LinTpNode	1	ref	The source of the TP connection.

Table 5.123: LinTpConnection

5.15.5 SAE J1939 Transport Layer

There are two transport protocols defined for J1939: BAM (Broadcast Announce Message), which is a broadcast protocol that does not use any flow control, and CMDT (Connection Mode Data Transfer), which is a point-to-point protocol with flow control.

BAM uses two `NPdus` for transport, TP.CM (Transport Protocol Command) and TP.DT (Transport Protocol Data). CMDT uses three `NPdus`, because an additional TP.CM in reverse direction is needed for flow control. The length of TP.CM and TP.DT NPdus is fixed to 8 bytes.

Both transport protocols can be described with the `J1939TpConfig` element. A `J1939TpConnection` defines the way of communication between a sender and a receiver and provides `NPdus` which are used to transmit TP.CM and TP.DT to the CAN Interface. If CMDT is used an additional reference to the `FlowControlNPdu` shall be defined. In case of variable length IPdus (with system signals of variable length), an additional NPdu (directNPdu) is used for messages with up to 8 bytes.

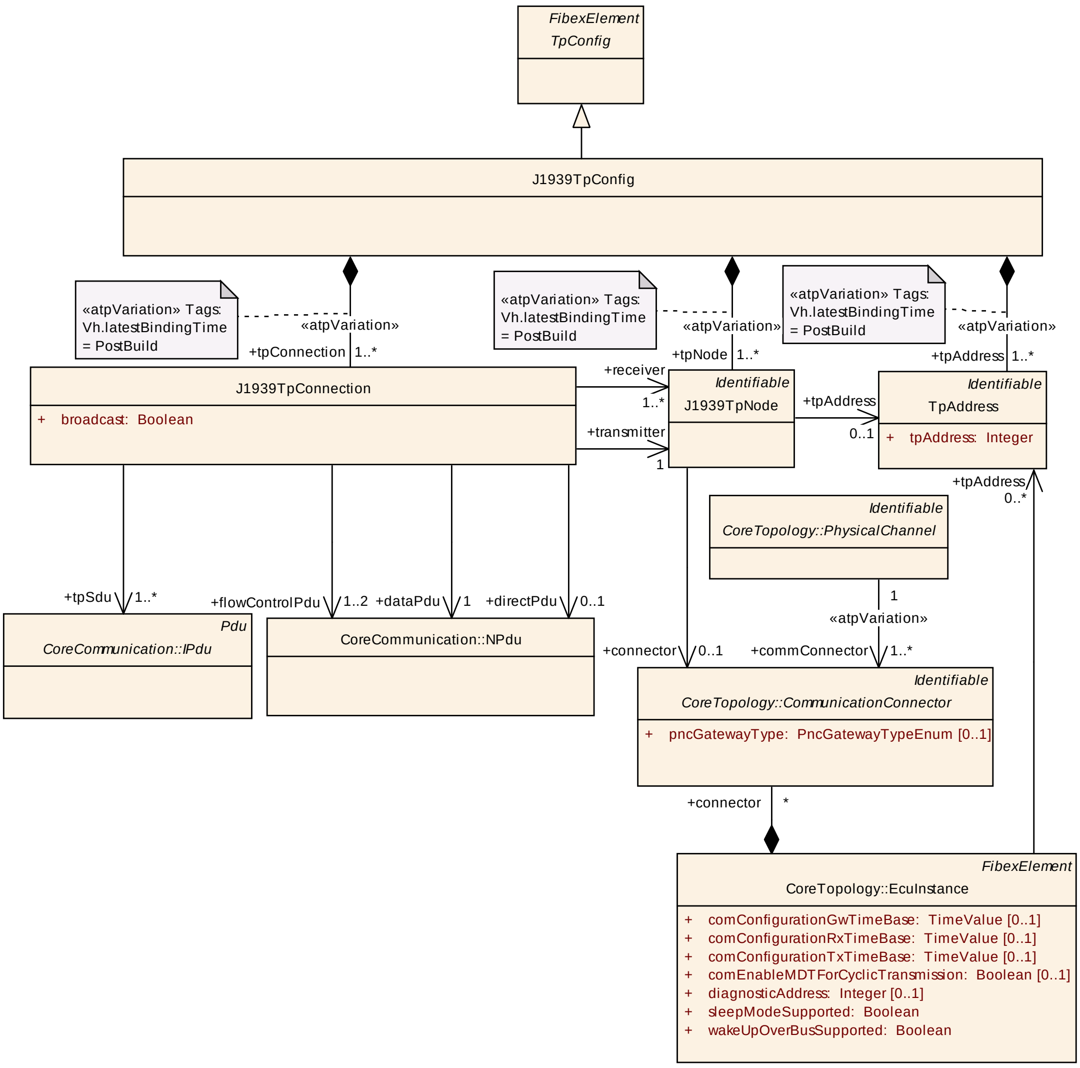

Figure 5.28: J1939 Transport Layer Configuration

A J1939TpConnection is specifically used for communication between one source and one or several target devices. These communication partners are specified using the source and target associations to J1939TpNode, providing the diagnostic tpAddress and the connection to the topology. BAM (Broadcast Announce Message), is always directed at the target address 0xFF, so there is no target address reference necessary for the broadcast situation.

The Parameter Group (PG) to be transported by the J1939TpConnection is specified by the reference tpSdu to IPdu.

Class	**J1939TpConfig**			
Package	M2::AUTOSARTemplates::SystemTemplate::TransportProtocols			
Note	This element defines exactly one J1939 TP Configuration. One J1939TpConfig element shall be created for each J1939 Network in the System. **Tags:** atp.recommendedPackage=TpConfigs			
Base	ARObject,CollectableElement,FibexElement,Identifiable,Multilanguage Referrable,PackageableElement,Referrable,TpConfig			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**
tpAddress	TpAddress	1..*	aggr	Collection of TP Adresses. atpVariation: Derived, because EcuInstance can vary. **Stereotypes:** atpVariation **Tags:** Vh.latestBindingTime=PostBuild
tpConnection	J1939TpConnection	1..*	aggr	Configuration of J1939 TP connections. atpVariation: Derived, because TpNode can vary. **Stereotypes:** atpVariation **Tags:** Vh.latestBindingTime=PostBuild
tpNode	J1939TpNode	1..*	aggr	Senders and receivers of J1939 TP messages. atpVariation: Derived, because EcuInstance can vary. **Stereotypes:** atpVariation **Tags:** Vh.latestBindingTime=PostBuild

Table 5.124: J1939TpConfig

Class	**J1939TpConnection**			
Package	M2::AUTOSARTemplates::SystemTemplate::TransportProtocols			
Note	A J1939TpConnection represents an internal path for the transmission or reception of a Pdu via J1939Tp and describes the the sender and the receiver of this particular communication. The J1939Tp module routes a Pdu (J1939 PGN) through the connection.			
Base	ARObject			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
broadcast	Boolean	1	attr	BAM (Broadcast Announce Message) is a broadcast protocol. If this attribute is set to true broadcast is used. Since address FF is the only broadcast address, there's no reason to configure it.
dataPdu	NPdu	1	ref	There are two transport protocols defined for J1939: BAM (Broadcast Announce Message), which is a broadcast protocol, and CMDT (Connection Mode Data Transfer), which is a point-to-point protocol with flow control. BAM uses one Data NPdu for transport, TP.DT (Transport Protocol Data) and one FlowControlNpdu, TP.CM (Transport Protocol Command). CMDT uses three CAN frames (one Data NPdu and two FlowControlNPdus) because an additional TP.CM in reverse direction is needed for flow control. The DataNPdu has a fixed length of 8 bytes.
directPdu	NPdu	0..1	ref	In case of variable length IPdus (with system signals of variable length), an additional NPdu (with the PGN in the CAN ID) is used for messages with up to 8 bytes.
flowControl Pdu	NPdu	1..2	ref	Reference to the Flow Control NPdus that are used in the CMDT (Connection Mode Data Transfer) for TP.CM in both directions. BAM uses one TP.CM (Transport Protocol Command). The flowControlNPdu has a fixed length of 8 bytes.
receiver	J1939TpNode	1..*	ref	The target of the TP connection.
tpSdu	IPdu	1..*	ref	Reference to IPdus that are segmented by the Transport Protocol. To support the low-level routing of NPdu's the NPdu is a specialization of an IPdu. More details can be found in the NPdu class description. Nevertheless the J1939TpConnection must not reference a NPdu with this tpSdu reference.
transmitter	J1939TpNode	1	ref	The source of the TP connection.

Table 5.125: J1939TpConnection

Class	**J1939TpNode**			
Package	M2::AUTOSARTemplates::SystemTemplate::TransportProtocols			
Note	TP Node (Sender or Receiver) provides the TP Address and the connection to the Topology description.			
Base	ARObject,Identifiable,MultilanguageReferrable,Referrable			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
connector	Communication Connector	0..1	ref	Asscociation to a CommunicationConnector in the topology description. In a System Description this reference is mandatory. In an ECU Extract this reference is optional (references to ECUs that are not part of the ECU Extract shall be avoided).
tpAddress	TpAddress	0..1	ref	Reference to the TP Address that is used by the TpNode. This reference is optional only when no TP is sent and only BAM is received.

Table 5.126: J1939TpNode

5.15.6 Unicast TP Example

The example in Figure 5.29 illustrate the usage of the System Template TP model. In this example the Sender ECU communicates with the Receiver ECU via two Gateways (GW1 and GW2).

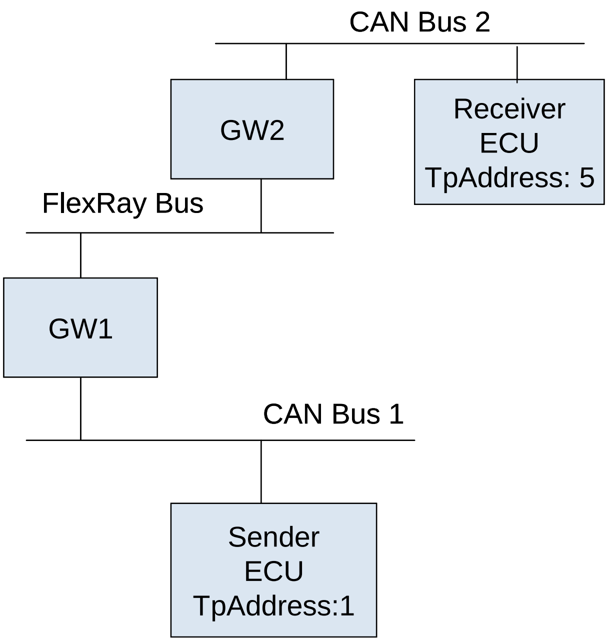

Figure 5.29: TP unicast Example

Modeling in the System Description:

```
CAN Bus 1 (CanTpConfig 1):
CanTpConnection
  transmitter TpNode: Sender ECU, TpAddress: 1
  receiver TpNode: GW1, TpAddress: 5

FlexRay Bus (FlexRayTpConfig):
FlexRayTpConnection
  transmitter TpNode: GW1, TpAddress: 1
  receiver TpNode: GW2, TpAddress: 5

CAN Bus 2 (CanTpConfig 2):
CanTpConnection
  transmitter TpNode: GW2, TpAddress: 1
  receiver TpNode: Receiver ECU, TpAddress: 5
```

Please note that two different `CanTpConfig` elements are created for the two CAN networks. The `TpAddress` of the transmitter `TpNode` is always 1 and the `TpAddress` of the receiver `TpNode` is always 5, even in the `FlexRayTpConfig` where Gateway ECU1 communicates with Gateway ECU2. The original transmitter and the final receiver are addressed in each connection.

5.15.7 Multicast TP Example

A second example illustrates the usage of the multicast reference.

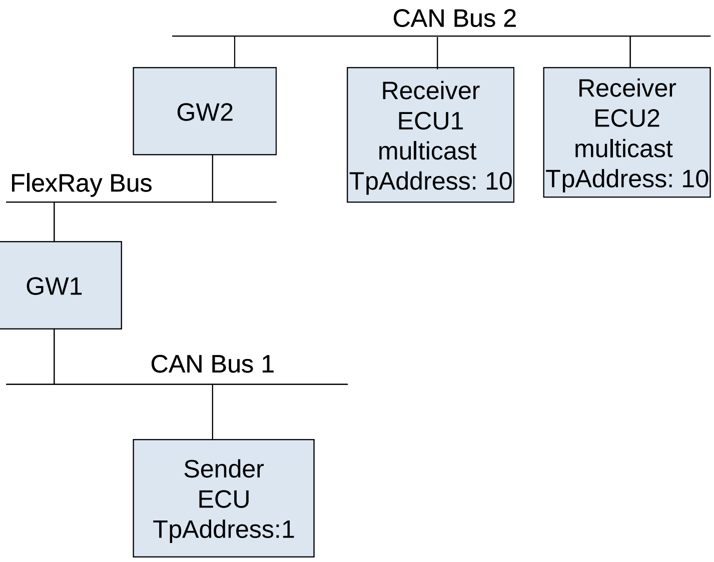

Figure 5.30: TP multicast Example

```
Can Bus 1 (CanTpConfig1):
CanTpConnection
  source TpNode: Sender ECU, TpAddress: 1
  target TpNode: GW1
  multicast TpAddress: 10

FlexRay Bus  (FlexRayTpConfig):
FlexRayTpConnection
  source TpNode: GW1, TpAddress: 1
  target TpNode: GW2
  multicast TpAddress: 10

CAN Bus 2 (CanTpConfig 2):
CanTpConnectionChannel
  source TpNode: GW2, TpAddress: 1
  target TpNode: Receiver ECU1
  target TpNode: Receiver ECU2
  multicast TpAddress: 10
```

Please note that the target `TpNode` does not contain a reference to the `TpAddress`. The multicast `TpAddress` is described by a direct reference from the connection.

5.16 Network Management

The Network Management in AUTOSAR is responsible for the cluster wide coordinated switching of the ECU nodes between operational modes (Network Mode, Synchronize Mode and Bus-Sleep Mode). The AUTOSAR NM coordination algorithm (FlexRay and CAN) is based on periodic NM-Vote messages received by all nodes in the cluster. Reception of an NM-Vote messages indicates that the sending node wants to keep the NM-cluster awake. If any node is ready to go to the Bus-Sleep Mode, it stops sending NMPdus, but as long as NMPdus from other nodes are received, it postpones transition to the Bus-Sleep Mode. Ultimately, if a designated timer elapses because no NMPdus are received anymore, the node initiates transition to the Bus-Sleep Mode.

The NM specification of AUTOSAR consist of a Generic Network Management Interface Module and of bus specific Network management adaptation layers (CanNm, FrNm, UdpNm). The AUTOSAR Generic NM Interface module acts as a bus-independent adaptation layer between the bus-specific Network Management modules and the AUTOSAR basic software module Communication Manager. Consequently, the peculiarities of the cluster types are adressed in the System Template by having different detailed models for FlexRay, CAN and Udp. However, all models are embedded into the communication model: They use specialized classes of NmConfig as a root element into the Nm configuration.

The parameters that are necessary to configure the Generic Network Management Interface Module are collected in the elements NMCluster, NMEcu, NM Coordinator and NMNode. See also figure 5.31.

The NMCluster contains a set of NMNodes that are coordinated with use of the NM algorithm. The NMNodes are associated with the CommunicationController in the topology and belong to exactly one NMEcu. The reception and transmission of NM messages is specified with the rxNmPdu and txNmPdu associations to NmPdus.

An NM Coordinator is an NMEcu, which is connected to at least two clusters ((via NmNodes), and where the requirement exists that shutdown of NM of at least two of these busses has to be performed synchronously.

Class	**NmConfig**			
Package	M2::AUTOSARTemplates::SystemTemplate::NetworkManagement			
Note	Contains the all configuration elements for AUTOSAR Nm. **Tags:** atp.recommendedPackage=NmConfigs			
Base	ARObject,CollectableElement,FibexElement,Identifiable,Multilanguage Referrable,PackageableElement,Referrable			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
nmCluster	NmCluster	1..*	aggr	Collection of NM Clusters atpVariation: Derived, because cluster can be variable. **Stereotypes:** atpVariation **Tags:** Vh.latestBindingTime=PostBuild
nmCluster Coupling	NmClusterCoup ling	1..*	aggr	Collection of NmClusterCouplings atpVariation: Derived, because NmCluster can vary. **Tags:** Vh.latestBindingTime=PostBuild
nmIfEcu	NmEcu	1..*	aggr	Collection of NM ECUs atpVariation: Derived, because EcuInstance can be variable. **Stereotypes:** atpVariation **Tags:** Vh.latestBindingTime=PreCompileTime

Table 5.127: NmConfig

Class	**NmCluster (abstract)**			
Package	M2::AUTOSARTemplates::SystemTemplate::NetworkManagement			
Note	Set of NM nodes coordinated with use of the NM algorithm.			
Base	ARObject,Identifiable,MultilanguageReferrable,Referrable			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
communic ationCluste r	Communication Cluster	1	ref	Asscociation to a CommunicationCluster in the topology description.
nmChanne lId	PositiveInteger	1	attr	Channel identification number of the corresponding channel. Must be unique over all NmClusters.
nmChanne lSleepMast er	Boolean	1	attr	This parameter shall be set to indicate if the sleep of this network can be absolutely decided by the local node only and that no other nodes can oppose that decision.

Attribute	Datatype	Mul.	Kind	Note
nmNode	NmNode	1..*	aggr	Collection of NmNodes of the NmCluster. atpVariation: Derived, because NmNode can be variable. **Stereotypes:** atpVariation **Tags:** Vh.latestBindingTime=PostBuild
nmSynchronizingNetwork	Boolean	1	attr	If this parameter is true, then this network is a synchronizing network for the NM coordination cluster which it belongs to. The network is expected to call Nm_SynchronizationPoint() at regular intervals.

Table 5.128: NmCluster

Class	NmEcu			
Package	M2::AUTOSARTemplates::SystemTemplate::NetworkManagement			
Note	ECU on which NM is running.			
Base	ARObject,Identifiable,MultilanguageReferrable,Referrable			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**
busSpecificNmEcu	BusspecificNmEcu	0..1	aggr	Busspecific NmEcu attributes
ecuInstance	EcuInstance	1	ref	Asscociation to an ECUInstance in the topology description.
nmBusSynchronizationEnabled	Boolean	0..1	attr	Enables bus synchronization support.
nmComControlEnabled	Boolean	0..1	attr	Enables the Communication Control support.
nmCoordinator	NmCoordinator	0..1	aggr	Nm ECU may coordinate different clusters.
nmCycletimeMainFunction	TimeValue	0..1	attr	The period between successive calls to the Main Function of the NM Interface in seconds.
nmNodeDetectionEnabled	Boolean	0..1	attr	Enables the Request Repeat Message Request support. Only valid if nmNodeIdEnabled is set to true.
nmNodeIdEnabled	Boolean	0..1	attr	Enables the source node identifier.
nmPassiveModeEnabled	Boolean	0..1	attr	This attribute is deprecated and shall not be used. It is only kept in the model for backward compatibility reasons and will be removed in the future. The passive mode is configurable per channel with the attribute nmPassiveModeEnabled in NmNode. **Tags:** atp.Status=obsolete
nmPduRxIndicationEnabled	Boolean	0..1	attr	Switch for enabling the PDU Rx Indication.

Attribute	Datatype	Mul.	Kind	Note
nmRemoteSleepIndEnabled	Boolean	0..1	attr	Switch for enabling remote sleep indication support.
nmRepeatMsgIndEnabled	Boolean	0..1	attr	Switch for enabling the Repeat Message Bit Indication.
nmStateChangeIndEnabled	Boolean	0..1	attr	Enables the CAN Network Management state change notification.
nmUserDataEnabled	Boolean	0..1	attr	Switch for enabling user data support.

Table 5.129: NmEcu

Class	NmCoordinator			
Package	M2::AUTOSARTemplates::SystemTemplate::NetworkManagement			
Note	A NM coordinator is an ECU, which is connected to at least two busses, and where the requirement exists that shutdown of NM of at least two of these busses (also referred to as coordinated busses) has to be performed synchronously.			
Base	ARObject			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**
index	Integer	1	attr	Identification of the NMCoordinator.
nmActiveCoordinator	Boolean	0..1	attr	This attribute indicates whether a NM Coordinator is an active gateway (true) or a passive gateway (false).
nmGlobalCoordinatorTime	TimeValue	0..1	attr	This attribute defines the maximum shutdown time (in seconds) of a connected and coordinated NM-Cluster.
nmNode	NmNode	1..*	ref	reference to busses (via NmNodes) that are coordinated by the NmCoordinator.
nmShutdownDelayTimer	TimeValue	1	attr	This parameter defines the time in seconds which the NM Coordination algorithm shall delay the release of the referenced cluster.

Table 5.130: NmCoordinator

Class	NmNode (abstract)			
Package	M2::AUTOSARTemplates::SystemTemplate::NetworkManagement			
Note	The linking of NmEcus to NmClusters is realized via the NmNodes.			
Base	ARObject,Identifiable,MultilanguageReferrable,Referrable			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**
controller	CommunicationController	1	ref	Asscociation to an CommunicationController in the topology description.
nmIfEcu	NmEcu	1	ref	Reference to the NmEcu that contains this NmNode. (CommunicationController that is referenced by the NmNode shall be contained in the EcuInstance that is referenced by the NmEcu).
nmNodeId	Integer	0..1	attr	Node identifier of local NmNode. Must be unique in the NmCluster.

Attribute	**Datatype**	**Mul.**	**Kind**	**Note**
nmPassiveModeEnabled	Boolean	0..1	attr	Enables support of the Passive Mode. The passive mode is configurable per channel.
rxNmPdu	NmPdu	1..*	ref	receive NM Pdu.
txNmPdu	NmPdu	*	ref	transmit NM Pdu

Table 5.131: NmNode

Class	**NmClusterCoupling (abstract)**			
Package	M2::AUTOSARTemplates::SystemTemplate::NetworkManagement			
Note	Attributes that are valid for each of the referenced (coupled) clusters.			
Base	ARObject			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**
–	–	–	–	–

Table 5.132: NmClusterCoupling

5.16.1 FlexRay Network Management

The following class tables specify the configuration parameters of FlexRay NM.

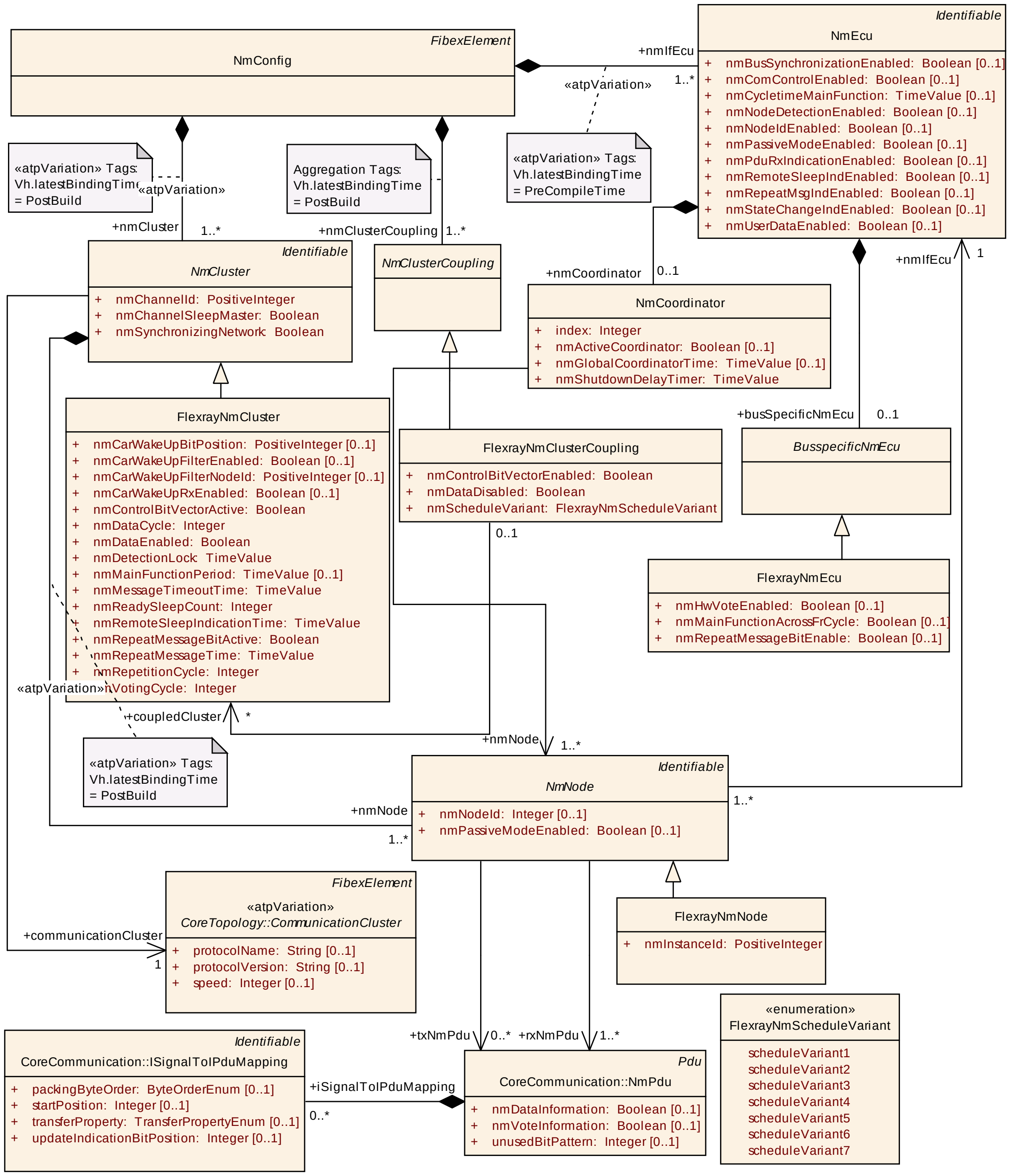

Figure 5.31: FlexRay Network Management Configuration (TransportProtocols: NmFlexRayConfiguration)

Class	FlexrayNmCluster			
Package	M2::AUTOSARTemplates::SystemTemplate::NetworkManagement			
Note	FlexRay specific NM cluster attributes.			
Base	ARObject,Identifiable,MultilanguageReferrable,NmCluster,Referrable			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
nmCarWakeUpBitPosition	PositiveInteger	0..1	attr	Specifies the bit position of the CarWakeUp within the NM-Message.
nmCarWakeUpFilterEnabled	Boolean	0..1	attr	If this attribute is set to true the CareWakeUp filtering is supported. In this case only the CarWakeUp bit within the NM message with source node identifier nmCarWakeUpFilterNodeId is considered as CarWakeUp request.
nmCarWakeUpFilterNodeId	PositiveInteger	0..1	attr	Source node identifier for CarWakeUp filtering. If CarWakeUp filtering is supported (nmCarWakeUpFilterEnabled), only the CarWakeUp bit within the NM message with source node identifier nmCarWakeUpFilterNodeId is considered as CarWakeUp request.
nmCarWakeUpRxEnabled	Boolean	0..1	attr	If set to true this attribute enables the support of CarWakeUp bit evaluation in received NM messages.
nmControlBitVectorActive	Boolean	1	attr	Used to activate or deactivate the control bit vector support for a Fr Nm Channel.
nmDataCycle	Integer	1	attr	Number of FlexRay Communication Cycles needed to transmit the Nm Data PDUs of all FlexRay Nm Ecus of this FlexRayNmCluster.
nmDataEnabled	Boolean	1	attr	Switch to enable the separated sending of NM-Data. True: enables False: disables
nmDetectionLock	TimeValue	1	attr	The time for which a node will not set the repeat message request bit even in the presence of a repeat message request (in seconds).
nmMainFunctionPeriod	TimeValue	0..1	attr	Defines the processing cycle of the main function of FrNm module.
nmMessageTimeoutTime	TimeValue	1	attr	Timeout of a NM message in seconds. It determines how long the NM shall wait with notification of transmission failure while communication errors occur on the bus.
nmReadySleepCount	Integer	1	attr	Numbers of repetitions in the ready sleep state before NM switches to bus sleep mode. On a value of "1", the NM-State Machine will leave the Ready Sleep State after one NM Repetition Cycle with no "keep awake" votes.
nmRemoteSleepIndicationTime	TimeValue	1	attr	Timeout for Remote Sleep Indication in seconds. It defines the time how long it shall take to recognize that all other nodes are ready to sleep.
nmRepeatMessageBitActive	Boolean	1	attr	Used to activate or deactivate the repeat message bit support for a Fr Nm Channel.

Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
nmRepeatMessageTime	TimeValue	1	attr	Timeout for Repeat Message State in seconds. Defines the time how long the NM shall stay in the Repeat Message State.
nmRepetitionCycle	Integer	1	attr	Number of FlexRay Communication Cycles used to repeat the transmission of the Nm vote PDUs of all FlexRay NmEcus of this FlexRayNmCluster. This value must be an integral multiple of nmVotingCycle.
nmVotingCycle	Integer	1	attr	Number of FlexRay CommunicationCycles needed to transmit the Nm vote of Pdus of all FlexRay NmEcus of this FlexRayNmCluster.

Table 5.133: FlexrayNmCluster

Class	**FlexrayNmEcu**			
Package	M2::AUTOSARTemplates::SystemTemplate::NetworkManagement			
Note	FlexRay specific attributes.			
Base	ARObject,BusspecificNmEcu			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
nmHwVoteEnabled	Boolean	0..1	attr	Switch for enabling the processing of FlexRay Hardware aggregated NM-Votes.
nmMainFunctionAcrossFrCycle	Boolean	0..1	attr	Parameter describing if the execution of the FrNm_Main function crosses theFlexRay cycle boundary or not.
nmRepeatMessageBitEnable	Boolean	0..1	attr	Enables/disables the repeat message bit support

Table 5.134: FlexrayNmEcu

Class	**FlexrayNmClusterCoupling**			
Package	M2::AUTOSARTemplates::SystemTemplate::NetworkManagement			
Note	FlexRay attributes that are valid for each of the referenced (coupled) FlexRay clusters.			
Base	ARObject,NmClusterCoupling			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
coupledCluster	FlexrayNmCluster	*	ref	Reference to coupled FlexRay Clusters.
nmControlBitVectorEnabled	Boolean	1	attr	Enables control bit vector support.
nmDataDisabled	Boolean	1	attr	Disables the transmission of NM-Data.
nmScheduleVariant	FlexrayNmScheduleVariant	1	attr	FrNm schedule variant according to FrNm SWS.

Table 5.135: FlexrayNmClusterCoupling

Class	**FlexrayNmNode**			
Package	M2::AUTOSARTemplates::SystemTemplate::NetworkManagement			
Note	FlexRay specific NM Node attributes.			
Base	ARObject,Identifiable,MultilanguageReferrable,NmNode,Referrable			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
nmInstanceId	PositiveInteger	1	attr	The NM instance identifier is used for reporting of development errors to DET. It must be unique for each NM instance within one ECU.

Table 5.136: FlexrayNmNode

Enumeration	**FlexrayNmScheduleVariant**
Package	M2::AUTOSARTemplates::SystemTemplate::NetworkManagement
Note	FrNm schedule variant according to FrNm SWS.
Literal	***Description***
scheduleVariant1	NM-Vote and NM Data transmitted within one PDU in static segment. The NM-Vote has to be realized as separate bit within the PDU.
scheduleVariant2	NM-Vote and NM-Data transmitted within one PDU in dynamic segment. The presence (or non-presence) of the PDU corresponds to the NM-Vote
scheduleVariant3	NM-Vote and NM-Data are transmitted in the static segment in separate PDUs. This alternative is not recommended => Alternative 1 should be used instead.
scheduleVariant4	NM-Vote transmitted in static and NM-Data transmitted in dynamic segment.
scheduleVariant5	NM-Vote is transmitted in dynamic and NM-Data is transmitted in static segment. This alternative is not recommended => Variants 2 or 6 should be used instead.
scheduleVariant6	NM-Vote and NM-Data are transmitted in dynamic segment in separate PDUs.
scheduleVariant7	NM-Vote and a copy of the CBV are transmitted in the static segment (using the FlexRay NM Vector support) and NM-Data is transmitted in the dynamic segment

Table 5.137: FlexrayNmScheduleVariant

— AUTOSAR CONFIDENTIAL —

5.16.2 CAN Network Management

The following class tables specify the configuration parameters of CAN NM.

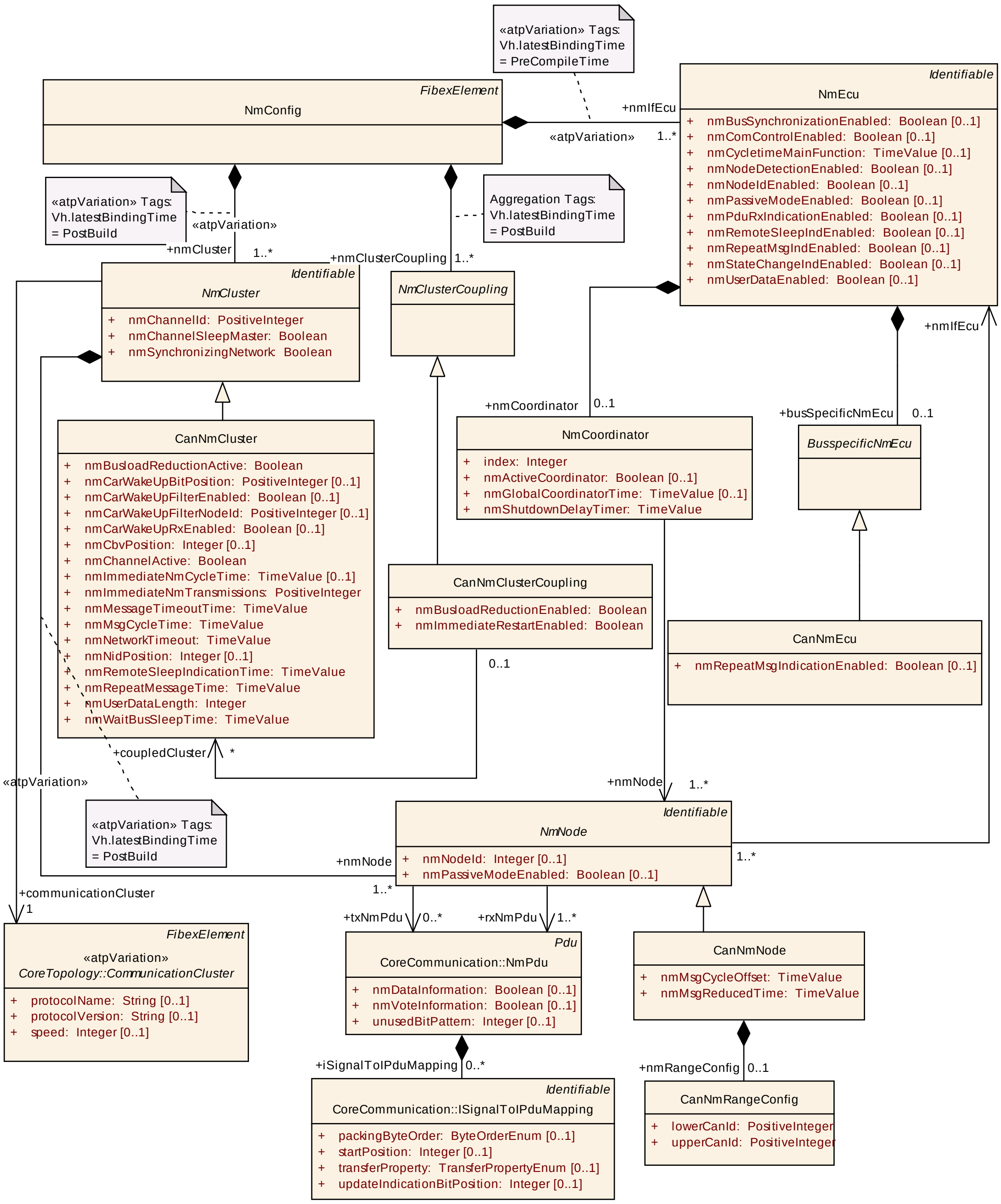

Figure 5.32: CAN Network Management Configuration (TransportProtocols: NmCanConfiguration)

Class	**CanNmCluster**			
Package	M2::AUTOSARTemplates::SystemTemplate::NetworkManagement			
Note	Can specific NmCluster attributes			
Base	ARObject,Identifiable,MultilanguageReferrable,NmCluster,Referrable			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**
nmBusloadReductionActive	Boolean	1	attr	It determines if bus load reduction for the respective CanNm channel is active or not.
nmCarWakeUpBitPosition	PositiveInteger	0..1	attr	Specifies the bit position of the CarWakeUp within the NM-Message.
nmCarWakeUpFilterEnabled	Boolean	0..1	attr	If this attribute is set to true the CareWakeUp filtering is supported. In this case only the CarWakeUp bit within the NM message with source node identifier nmCarWakeUpFilterNodeId is considered as CarWakeUp request.
nmCarWakeUpFilterNodeId	PositiveInteger	0..1	attr	Source node identifier for CarWakeUp filtering. If CarWakeUp filtering is supported (nmCarWakeUpFilterEnabled), only the CarWakeUp bit within the NM message with source node identifier nmCarWakeUpFilterNodeId is considered as CarWakeUp request.
nmCarWakeUpRxEnabled	Boolean	0..1	attr	If set to true this attribute enables the support of CarWakeUp bit evaluation in received NM messages.
nmCbvPosition	Integer	0..1	attr	Defines the position of the control bit vector within the NM PDU (Byte positon).
nmChannelActive	Boolean	1	attr	This switch determines if the respective CanNm channel is active or not. Indicates whether a particular CanNm channel shall be initialized (TRUE) or not (FALSE). If this parameter is set to FALSE the respective NM instance shall not be used during runtime.
nmImmediateNmCycleTime	TimeValue	0..1	attr	Defines the immediate NM PDU cycle time in seconds which is used for nmImmediateNmTransmissions NM PDU transmissions. This parameter is only valid if CanNmImmediateNmTransmissions is greater one.
nmImmediateNmTransmissions	PositiveInteger	1	attr	Defines the number of immediate NM PDUs which shall be transmitted. If the value is zero no immediate NM PDUs are transmitted. The cycle time of immeditate NM PDUs is defined by nmImmediateNmCycleTime.
nmMessageTimeoutTime	TimeValue	1	attr	Timeout of a NM message in seconds. It determines how long the NM shall wait with notification of transmission failure while communication errors occur on the bus.

Attribute	Datatype	Mul.	Kind	Note
nmMsgCycleTime	TimeValue	1	attr	Period of a CanNm message in seconds. It determines the periodic rate in the periodic transmission mode with bus load reduction and is the basis for transmit scheduling in the periodic transmission mode without bus load reduction.
nmNetworkTimeout	TimeValue	1	attr	Network Timeout for CanNm PDUs in seconds It denotes the time how long the CanNm shall stay in the Network Mode before transition into Prepare Bus-Sleep Mode shall take place.
nmNidPosition	Integer	0..1	attr	Defines the byte position of the source node identifier within the NM PDU.
nmRemoteSleepIndicationTime	TimeValue	1	attr	Timeout for Remote Sleep Indication in seconds. It defines the time how long it shall take to recognize that all other nodes are ready to sleep.
nmRepeatMessageTime	TimeValue	1	attr	Timeout for Repeat Message State in seconds. Defines the time how long the NM shall stay in the Repeat Message State.
nmUserDataLength	Integer	1	attr	Defines the length of the user data contained in the NM Pdu.
nmWaitBusSleepTime	TimeValue	1	attr	Timeout for bus calm down phase in seconds. It denotes the time how long the CanNm shall stay in the Prepare Bus-Sleep Mode before transition into Bus-Sleep Mode shall take place.

Table 5.138: CanNmCluster

Class	**CanNmEcu**			
Package	M2::AUTOSARTemplates::SystemTemplate::NetworkManagement			
Note	CAN specific attributes.			
Base	ARObject,BusspecificNmEcu			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
nmRepeatMsgIndicationEnabled	Boolean	0..1	attr	Enable/disable the notification that a RepeatMessageRequest bit has been received. This attribute is deprecated and shall be not used. It will be removed in the future. The nmRepeatMsgIndEnabled attribute in NmEcu shall be used instead. **Tags:** atp.Status=obsolete

Table 5.139: CanNmEcu

Class	**CanNmClusterCoupling**			
Package	M2::AUTOSARTemplates::SystemTemplate::NetworkManagement			
Note	CAN attributes that are valid for each of the referenced (coupled) CAN clusters.			
Base	ARObject,NmClusterCoupling			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
coupledCluster	CanNmCluster	*	ref	Reference to coupled CAN Clusters.

Attribute	Datatype	Mul.	Kind	Note
nmBusload Reduction Enabled	Boolean	1	attr	Enables busload reduction support
nmImmedi ateRestart Enabled	Boolean	1	attr	Enables the asynchronous transmission of a CanNm PDU upon bus-communication request in Prepare-Bus-Sleep mode.

Table 5.140: CanNmClusterCoupling

Class	CanNmNode			
Package	M2::AUTOSARTemplates::SystemTemplate::NetworkManagement			
Note	CAN specific NM Node attributes.			
Base	ARObject,Identifiable,MultilanguageReferrable,NmNode,Referrable			
Attribute	Datatype	Mul.	Kind	Note
nmMsgCyc leOffset	TimeValue	1	attr	Node specific time offset in the periodic transmission node. It determines the start delay of the transmission. Specified in seconds.
nmMsgRe ducedTime	TimeValue	1	attr	Node specific bus cycle time in the periodic transmission mode with bus load reduction. Specified in seconds.
nmRange Config	CanNmRangeC onfig	0..1	aggr	Defines the CANID ranges that are used for Nm. This range definition is redundant to the attribute "rxIdentifierRange" of CanFrameTriggering. For backward compatibility reasons this redundancy shall be preserved and both shall be defined. In future this element will be removed from the model.

Table 5.141: CanNmNode

Class	CanNmRangeConfig			
Package	M2::AUTOSARTemplates::SystemTemplate::NetworkManagement			
Note	Defines the CANID ranges that are used for Nm. This range definition is redundant to the attribute "rxIdentifierRange" of CanFrameTriggering. For backward compatibility reasons this redundancy shall be preserved and both shall be defined. In future this element will be removed from the model. **Tags:** atp.Status=obsolete			
Base	ARObject			
Attribute	Datatype	Mul.	Kind	Note
lowerCanI d	PositiveInteger	1	attr	Lower CAN Identifier of a receive CAN L-PDU for identifier range definition.
upperCanI d	PositiveInteger	1	attr	Upper CAN Identifier of a receive CAN L-PDU for identifier range definition.

Table 5.142: CanNmRangeConfig

5.16.3 LIN Network Management

No relevant system information is described in the LinNm configuration. In AUTOSAR there is no communication between LinNm and LinIf and there are no dedicated LIN NM frames. Therefore a LinNm model in the System Template is unnecessary.

5.16.4 UDP Network Management

The following class tables specify the configuration parameters of UDP NM.

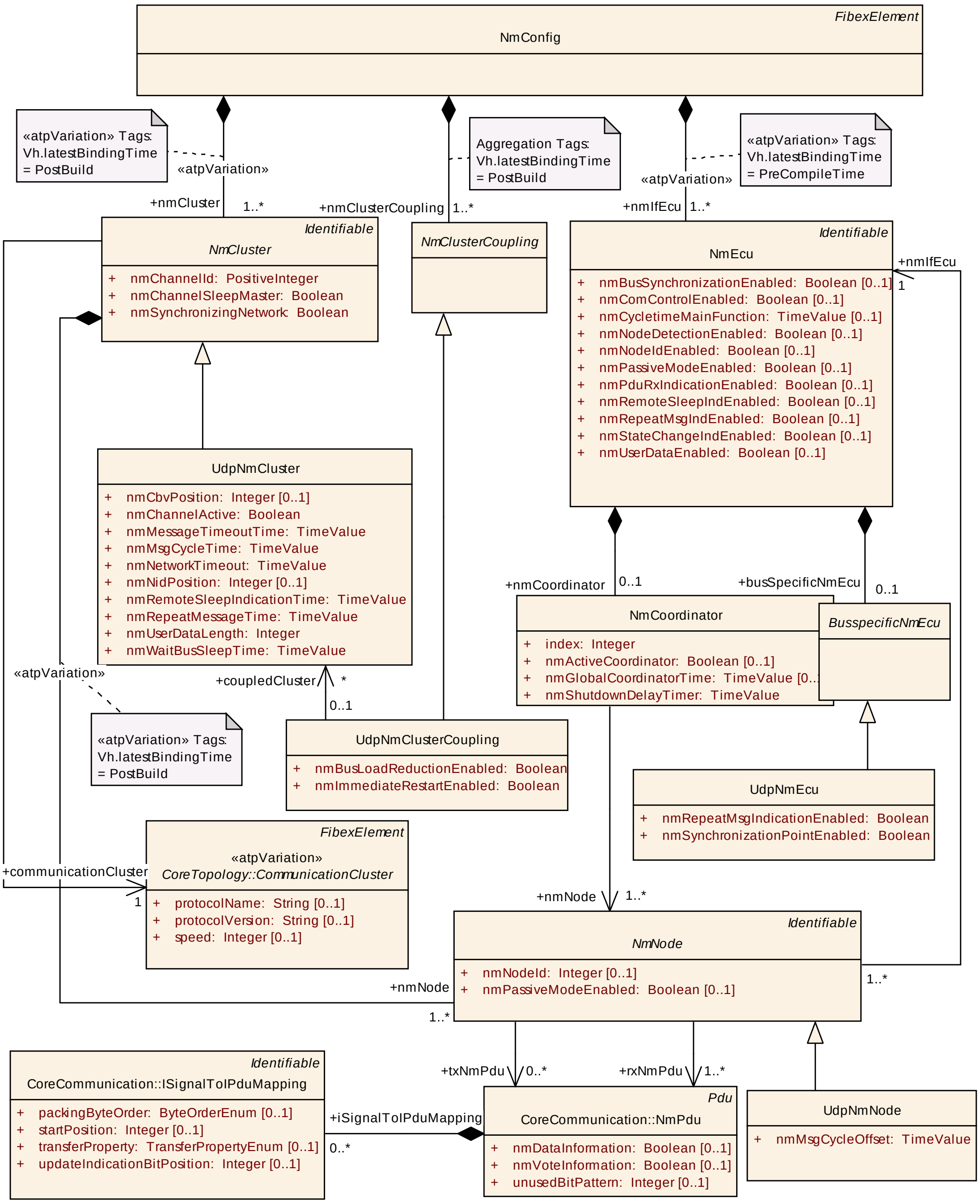

Figure 5.33: UDP Network Management Configuration (TransportProtocols: NmUdpConfiguration)

Class	**UdpNmCluster**			
Package	M2::AUTOSARTemplates::SystemTemplate::NetworkManagement			
Note	Udp specific NmCluster attributes			
Base	ARObject,Identifiable,MultilanguageReferrable,NmCluster,Referrable			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**
nmCbvPosition	Integer	0..1	attr	Defines the position of the control bit vector within the NM PDU (Byte positon).
nmChannelActive	Boolean	1	attr	This switch determines if the respective UdpNm channel is active or not. Indicates whether a particular UdpNm channel shall be initialized (TRUE) or not (FALSE). If this parameter is set to FALSE the respective NM instance shall not be used during runtime.
nmMessageTimeoutTime	TimeValue	1	attr	Timeout of a NM message in seconds. It determines how long the NM shall wait with notification of transmission failure while communication errors occur on the bus.
nmMsgCycleTime	TimeValue	1	attr	Period of a UdpNm message in seconds. It determines the periodic rate in the periodic transmission mode with bus load reduction and is the basis for transmit scheduling in the periodic transmission mode without bus load reduction.
nmNetworkTimeout	TimeValue	1	attr	Network Timeout for UdpNm PDUs in seconds. It denotes the time how long the CanNm shall stay in the Network Mode before transition into Prepare Bus-Sleep Mode shall take place.
nmNidPosition	Integer	0..1	attr	Defines the byte position of the source node identifier within the NM PDU.
nmRemoteSleepIndicationTime	TimeValue	1	attr	Timeout for Remote Sleep Indication in seconds. It defines the time how long it shall take to recognize that all other nodes are ready to sleep.
nmRepeatMessageTime	TimeValue	1	attr	Timeout for Repeat Message State in seconds. Defines the time how long the NM shall stay in the Repeat Message State.
nmUserDataLength	Integer	1	attr	Defines the length of the user data contained in the NM Pdu.
nmWaitBusSleepTime	TimeValue	1	attr	Timeout for bus calm down phase in seconds. It denotes the time how long the CanNm shall stay in the Prepare Bus-Sleep Mode before transition into Bus-Sleep Mode shall take place.

Table 5.143: UdpNmCluster

Class	**UdpNmEcu**			
Package	M2::AUTOSARTemplates::SystemTemplate::NetworkManagement			
Note	Udp NM specific ECU attributes.			
Base	ARObject,BusspecificNmEcu			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**
nmRepeatMsgIndicationEnabled	Boolean	1	attr	Enable/disable the notification that a RepeatMessageRequest bit has been received.
nmSynchronizationPointEnabled	Boolean	1	attr	Enable/disable the NM Coordination algorithm to being able to initiate the synchronization algorithm.

Table 5.144: UdpNmEcu

Class	**UdpNmClusterCoupling**			
Package	M2::AUTOSARTemplates::SystemTemplate::NetworkManagement			
Note	Udp attributes that are valid for each of the referenced (coupled) UdpNm clusters.			
Base	ARObject,NmClusterCoupling			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**
coupledCluster	UdpNmCluster	*	ref	Reference to coupled UdpNm Clusters.
nmBusLoadReductionEnabled	Boolean	1	attr	Enables busload reduction support
nmImmediateRestartEnabled	Boolean	1	attr	Enables the asynchronous transmission of a CanNm PDU upon bus-communication request in Prepare-Bus-Sleep mode.

Table 5.145: UdpNmClusterCoupling

Class	**UdpNmNode**			
Package	M2::AUTOSARTemplates::SystemTemplate::NetworkManagement			
Note	Udp specific NM Node attributes.			
Base	ARObject,Identifiable,MultilanguageReferrable,NmNode,Referrable			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**
nmMsgCycleOffset	TimeValue	1	attr	Node specific time offset in the periodic transmission node. It determines the start delay of the transmission. Specified in seconds.

Table 5.146: UdpNmNode

5.17 Fan-out

The RTE supports a "signal fan-out" where the same signal (System Signal) is sent in different IPdus to multiple receivers. The Pdu Router supports the "PDU fan-out" where the same IPdu is sent to multiple destinations. In COM the Signal Gateway supports a fan-out where an incoming signal is routed to several destinations. And the FlexRay interface supports a fan-out where the same Pdu is mapped into more than one frame.

5.17.1 RTE fan-out

- The RTE fan-out (signal fan-out) is described by the relation between SystemSignal and ISignal.
- In the case of a "signal fan-out", several ISignals in different IPdus refer to the same SystemSignal (see example in Figure 5.34).

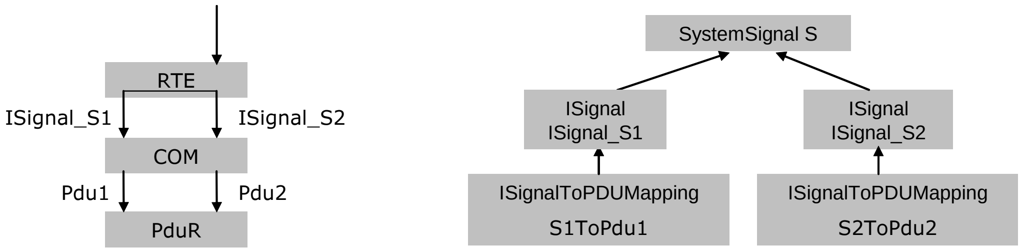

Figure 5.34: RTE fan-out

5.17.2 COM Signal Gateway fan-out

The COM Signal Gateway fan-out (1:n routing) is described with the definition of several `ISignalMappings` in the `Gateway` description, which all refer to the same source `ISignalTriggering`. All `ISignalTriggerings` (source and all destinations) that contribute to this Signal Mapping shall refer to the same `ISignal` since no RTE fan-out is provided by the COM Signal Gateway. The referenced `ISignal` is mapped into several `ISignalIPdus` (one for the source Signal and one for each destination signal).

5.17.3 Pdu Router fan-out

- The Pdu Router fan-out is described by the PduTriggering.[4] The sending ECU/PDU router has an output CommConnectorPort associated with the PduTriggering.

[4]The Pdu routing by the PduR is only allowed for IPdus and not for NmPdus and XcpPdus.

- According to the Cluster/Channel aggregation, the PDU-Router determines the clusters to use in its routing.
- The same IPdu is only sent once to each Bus Interface per Cluster: If PduTriggerings exist for more than one channel belonging to the same Cluster, the PDU Router still sends only one PDU transmission request to the bus Interface.

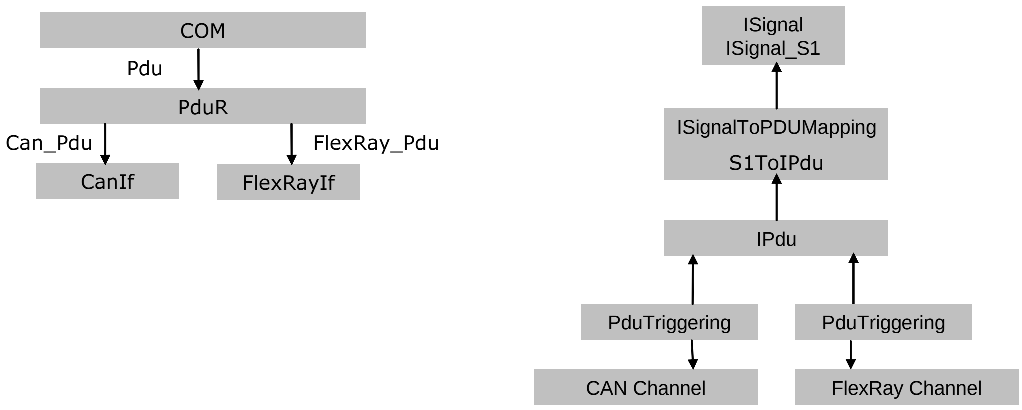

Figure 5.35: Pdu Router fan-out

5.17.4 Bus Interface fan-out

- The fan-out done in the FlexRay interface is described by the FrameTriggering element (The same Frame with the same Pdu content is transmitted over FlexRay channel A and FlexRay channel B, see example in Figure 5.36).
- There shall be a clear separation of responsibilities between PDU router and Flexray interface for handling PDU fan-out. This is further specified by the semantic rules on the Bus Interface below.
- If several frame triggerings with the same direction exist on the same cluster then the interface should handle the fan-out/in. In AUTOSAR frame routing is not supported.

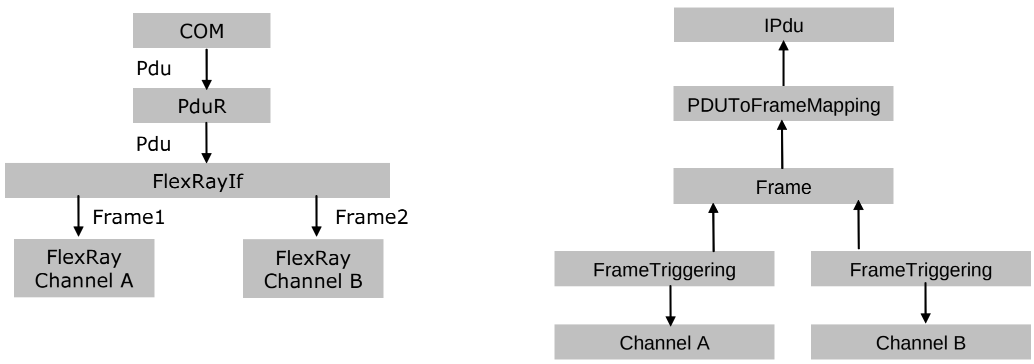

Figure 5.36: Bus Interface fan-out

5.17.5 Semantic Rules

- PduTriggering
  - Depending on its relation to entities such channels and clusters it can be unambiguously deduced whether a fan-out is handled by the Pdu Router or the Bus Interface.
  - If the fan-out is specified between different clusters it shall be handled by the Pdu Router.
  - If the fan-out is specified between different channels of the same cluster it shall be handled by the Bus Interface.
- FrameTriggering
  - For the same frame, if Frame Triggerings with the same direction exist on more than one channel of the same cluster the fan-out/in is handled by the interface.
- IPduToFrameMapping
  - Depending on its relation to entities such channels and clusters it can be unambiguously deduced whether a fan-out is handled by the Pdu router or the Bus Interface.
- Bus Interface
  - The Bus Interface does NOT handle fan-out/in between different clusters.

— AUTOSAR CONFIDENTIAL —

5.18 Support of Complex Device Drivers

The System Template allows the integration of custom communication means into AUTOSAR ECUs. The elements `UserDefinedPdu` and `UserDefinedIPdu` can be used to describe the Pdu-based communication via Complex Device Drivers. These elements are described in chapter 5.4 in more detail.

The `UserDefinedIPdu` can be used to describe the communication if a new BSW module was added above the PduR, e.g a Diagnostic Service.

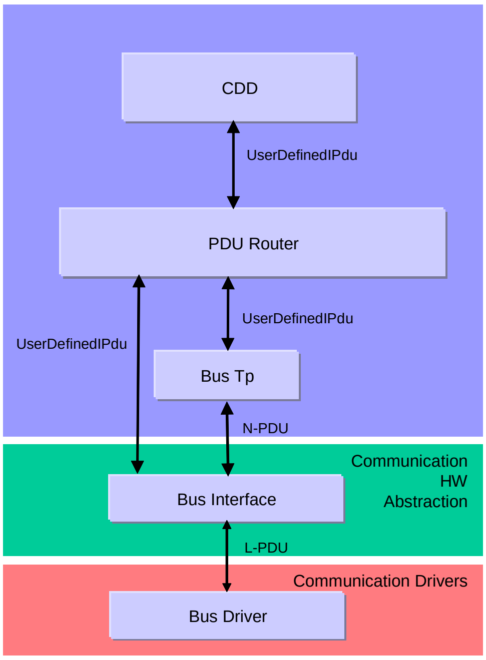

Figure 5.37: CDD over PduR

The `UserDefinedPdu` can be used to describe the communication if a new BSW module was added above an Interface, e.g. a new Nm module or XCP. A custom TP module can not be introduced since a CDD module can not be configured in the ECU Configuration as a lower layer of the Pdu Router. See [20] for more details.

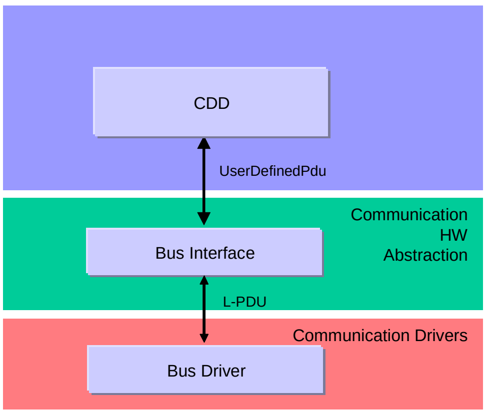

Figure 5.38: CDD over Bus Interface

6 Gateways

A gateway is a function within an ECU that performs as a Frame, I-Pdu or signal mapping function between two or more communication clusters.

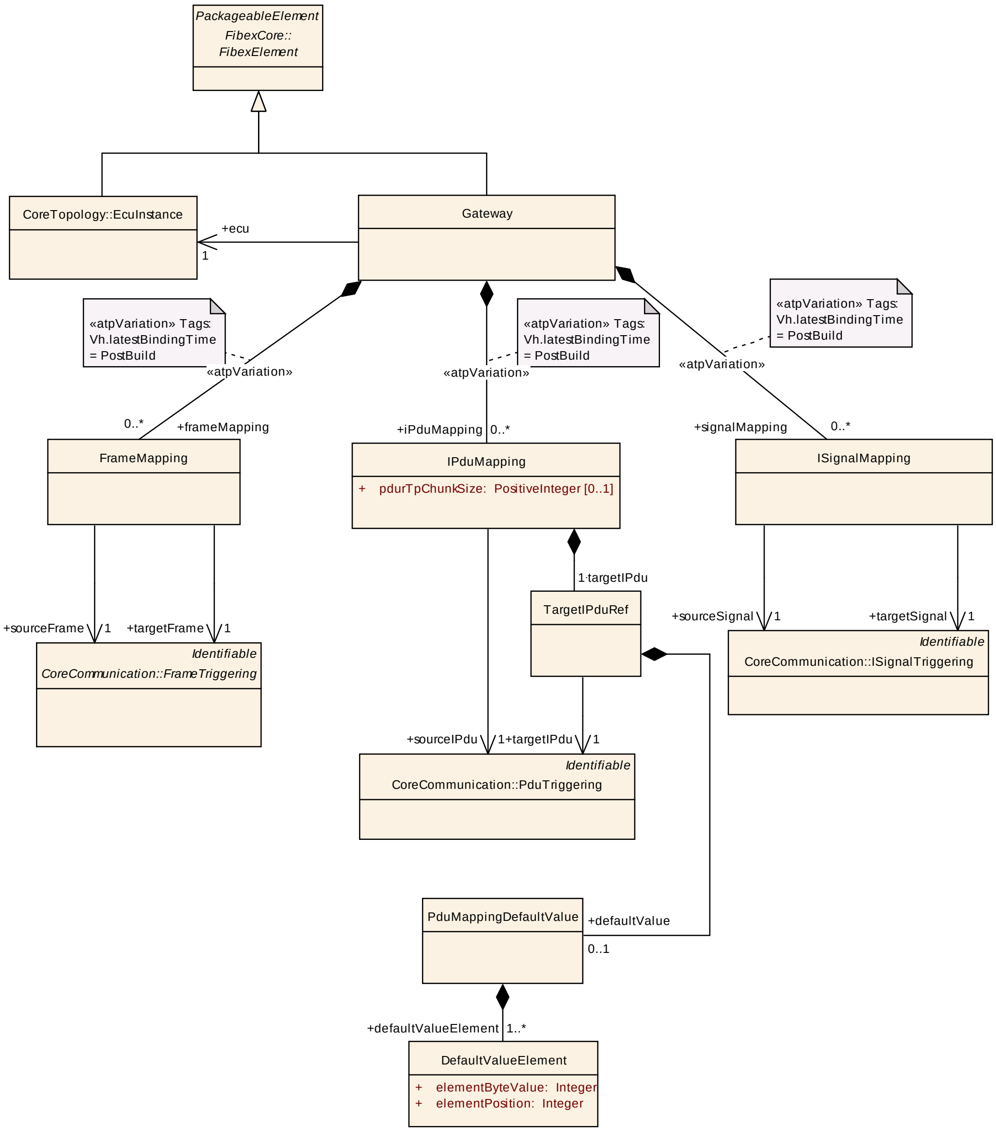

Figure 6.1: Communication Overview (Fibex4Multiplatform: Gateway)

Figure 6.1 shows the meta-model for the Gateway description in the System Template. It contains the following mapping functions:

- Frame Mapping
- I-Pdu Mapping
- Signal Mapping

Class	**Gateway**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Multiplatform			
Note	A gateway is an ECU that is connected to two or more clusters (channels, but not redundant), and performs a frame, Pdu or signal mapping between them. **Tags:** atp.recommendedPackage=Gateways			
Base	ARObject,CollectableElement,FibexElement,Identifiable,Multilanguage Referrable,PackageableElement,Referrable			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**
ecu	EcuInstance	1	ref	Reference to one ECU instance that implements the gateway.
frameMapping	FrameMapping	*	aggr	Frame Gateway: The entire source frame is mapped as it is onto the target frame (what in general is only possible inside of a common platform). In this case source and target frame should be the identical object. atpVariation: If frames are variable in clusters, the gateway frame mapping needs to be variable, too. **Stereotypes:** atpVariation **Tags:** Vh.latestBindingTime=PostBuild
iPduMapping	IPduMapping	*	aggr	IPdu Gateway: Arranges those IPdus that are transferred by the gateway from one channel to the other in pairs and defines the mapping between them. atpVariation: If PDUs are variable in clusters, the gateway PDU mapping needs to be variable, too. **Stereotypes:** atpVariation **Tags:** Vh.latestBindingTime=PostBuild
signalMapping	ISignalMapping	*	aggr	Signal Gateway: Arranges those signals that are transferred by the gateway from one channel to the other in pairs and defines the mapping between them. atpVariation: If signals are variable in clusters, the gateway signal mapping needs to be variable, too. **Stereotypes:** atpVariation **Tags:** Vh.latestBindingTime=PostBuild

Table 6.1: Gateway

— AUTOSAR CONFIDENTIAL —

6.1 Frame Mapping

The `FrameMapping` arranges those frames that are transferred by the gateway from one channel to the other in pairs and defines the mapping between them. Each pair consists in a Source and a Target referencing to a `FrameTriggering`.

The Frame Mapping is not supported by the AUTOSAR BSW. The existence is optional and has been incorporated into the System Template mainly for compatibility in order to allow interchange between FIBEX and AUTOSAR descriptions.

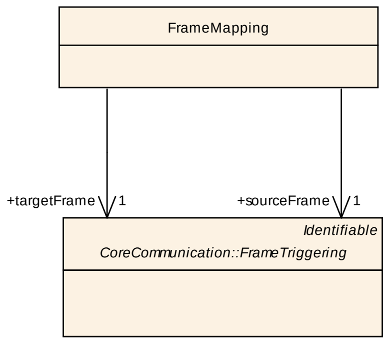

Figure 6.2: Frame Mapping (Fibex4Multiplatform: FrameMapping)

Class	**FrameMapping**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Multiplatform			
Note	The entire source frame is mapped as it is onto the target frame (what in general is only possible inside of a common platform). In this case source and target frame should be the identical object. Each pair consists in a SOURCE and a TARGET referencing to a FrameTriggering. The Frame Mapping is not supported by the Autosar BSW. The existence is optional and has been incorporated into the System Template mainly for compatibility in order to allow interchange between FIBEX and AUTOSAR descriptions.			
Base	ARObject			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
introduction	DocumentationBlock	0..1	aggr	This represents introductory documentation about the frame mapping.
sourceFrame	FrameTriggering	1	ref	Source destination of the referencing mapping.
targetFrame	FrameTriggering	1	ref	Target destination of the referencing mapping.

Table 6.2: FrameMapping

6.2 I-Pdu Mapping

The `IPduMapping` arranges those I-Pdus that are transferred by the gateway from one channel to the other in pairs and defines the mapping between them. Each pair consist of a source and a target referencing to a `PduTriggering`.

In the case that a Pdu is being gatewayed to more than one channel of the same cluster, all of this gateway relationships shall be specified. Therefore, all affected `PduTriggerings` must be described as gateway mappings.

The 1:n multicast routing is supported with the definition of several `IPduMappings`.

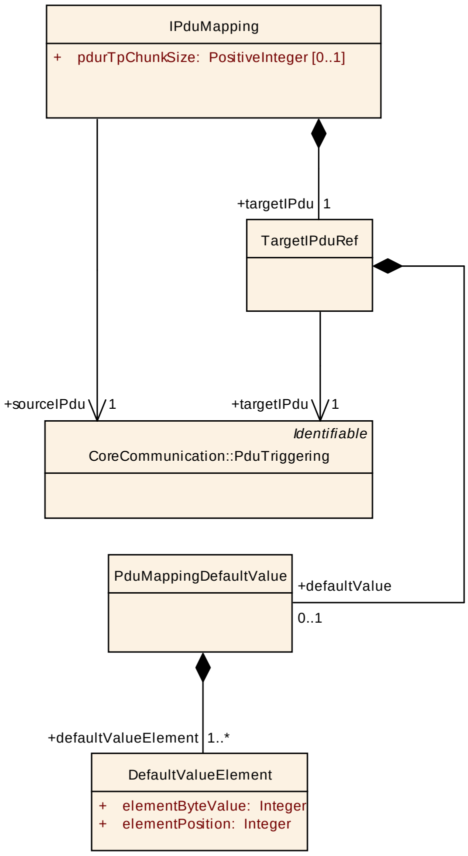

Figure 6.3: I-Pdu Mapping (Fibex4Multiplatform: IPduMapping)

Class	**IPduMapping**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Multiplatform			
Note	Arranges those IPdus that are transferred by the gateway from one channel to the other in pairs and defines the mapping between them.			
Base	ARObject			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**
introduction	DocumentationBlock	0..1	aggr	This represents introductory documentation about the IPdu mapping.
pdurTpChunkSize	PositiveInteger	0..1	attr	Optionally defines the to be configured Pdu Router TpChunkSize for this routing relation.
sourceIPdu	PduTriggering	1	ref	Source destination of the referencing mapping.
targetIPdu	TargetIPduRef	1	aggr	Target destination of the referencing mapping.

Table 6.3: IPduMapping

Class	**TargetIPduRef**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Multiplatform			
Note	Target destination of the referencing mapping.			
Base	ARObject			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**
defaultValue	PduMappingDefaultValue	0..1	aggr	If no I-Pdu has been received a default value will be distributed.
targetIPdu	PduTriggering	1	ref	IPdu Reference

Table 6.4: TargetIPduRef

Class	**PduMappingDefaultValue**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Multiplatform			
Note	Default Value which will be distributed if no I-Pdu has been received since last sending.			
Base	ARObject			
Attribute	**Datatype**	**Mul.**	**Kind**	**Note**
defaultValueElement	DefaultValueElement	1..*	aggr	The default value consists of a number of elements. Each default value element is represented by the element and the position in an array.

Table 6.5: PduMappingDefaultValue

Class	**DefaultValueElement**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Multiplatform			
Note	The default value consists of a number of elements. Each element is one byte long and the number of elements is specified by SduLength.			
Base	ARObject			
Attribute	*Datatype*	*Mul.*	*Kind*	*Note*
elementByteValue	Integer	1	attr	The integer value of a freely defined data byte.
elementPosition	Integer	1	attr	This attribute specifies the byte position of the element within the default value

Table 6.6: DefaultValueElement

6.3 Signal Mapping

The `ISignalMapping` arranges those signals and signal groups that are transferred by the gateway from one channel to the other in pairs and defines the mapping between them. Each pair consists of a source and a target referencing to an `ISignalTriggering`. Each `ISignalTriggering` points to either an `ISignal` or an `ISignalGroup` which are part of an `ISignalIPdu`. The `ISignal` refers to the to be routed `SystemSignal`, the `ISignalGroup` refers to the to be routed `SystemSignalGroup`.

The routing of a signal group is specified by defining the routing of the `ISignalGroup` pointing to the `SystemSignalGroup` (signal group). Routing specifications for `ISignalTriggerings` for group signals (`ISignals` contained in the `ISignalGroup`) shall not be defined. When performing a signal group routing the pairing of the `ISignals` is done by the `ISignal` reference from `ISignalGroup` to `ISignal`.

The 1:n multicast routing is supported with the definition of several `ISignalMappings`. See also the COM Signal Gateway fan-out description in section 5.17.2.

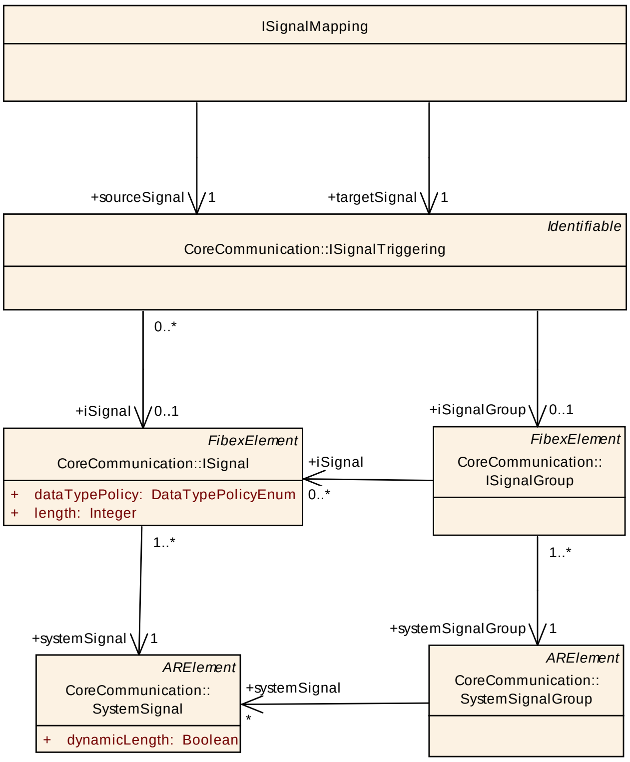

Figure 6.4: Signal Mapping (Fibex4Multiplatform: Signal Mapping)

Class	**ISignalMapping**			
Package	M2::AUTOSARTemplates::SystemTemplate::Fibex::Fibex4Multiplatform			
Note	Arranges those signals (or SignalGroups) that are transferred by the gateway from one channel to the other in pairs and defines the mapping between them. Each pair consists in a source and a target referencing to a ISignalTriggering.			
Base	ARObject			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
introduction	DocumentationBlock	0..1	aggr	This represents introductory documentation about the ISignal mapping.
sourceSignal	ISignalTriggering	1	ref	Source destination of the referencing mapping.
targetSignal	ISignalTriggering	1	ref	Target destination of the referencing mapping.

Table 6.7: ISignalMapping

6.3.1 Partial Signal Group Mapping

The signal mapping does support partial routing between signal groups which have not identical set of group signals. All group signals of the target signal group shall have a corresponding group signal from the source signal group. The partial routing of a signal group has to be performed in a way that the target signal group's content is consistent.

7 Usage of the System Template

As introduced in chapter 1.3 the System Template is used to describe the System Constraint Description, that serves as input to the AUTOSAR System Configuration Generator, and the System Configuration Description, that defines the output of the AUTOSAR System Configuration Generator. Certain elements of the System Template have a different meaning at the two stages of the AUTOSAR Methodology. The following table describes the differences of the elements.

Meta-classes, Chapters	*Usage to describe the System Contraints*	*Usage to describe the System Configuration*
`Topology` (2)	The Topology is completely described in the System Constraint Description.	The Topology description will be unchanged copied to the System Configuration description. The Topology may only be changed during another iteration development step of the whole system.

Meta-classes, Chapters (cont.)	*Usage to describe the System Contraints (cont.)*	*Usage to describe the System Configuration (cont.)*
`Communication` (5)	The System Constraint Description describes all frames that are predefined on all communication clusters of a vehicle. The predefinition of the communication matrix forces the system generator to use the given frame structure. Constraints for the system generator arise here e.g. from the used bus bandwidth, used identifiers as well as from the timing and at which position in a frame a Pdu is transmitted on the channel. Such a manual definition of the communication can be made for any reason where it is necessary to restrict the system generator. One example is the usage of legacy ECUs in an AUTOSAR System. The frames that are transmitted or received by these legacy ECUs are constraints for the system generator because they cannot be changed, if the compatibility is supposed to be achieved without any changes at the legacy ECUs.	In contrary to the System Constraint Description the final System Configuration Description contains all frames, Pdus and signals that will be sent by any ECU in the car. No matter if they were predefined (system constraint) or if they were generated by the system generator. The available information, in addition to the information, which is inserted by the AUTOSAR ECU configuration generator step, will be used as input to configure the Basic SW for the communication.

Meta-classes, Chapters (cont.)	*Usage to describe the System Contraints (cont.)*	*Usage to describe the System Configuration (cont.)*
`Gateway` (6)	The System Constraint Description describes all gateways in the system including their gateway entries that are predefined. The predefinition of the gateways or parts of the gateways can be used to define manually the copying of Frames, I-Pdus or signals. The reasons for such predefinitions are quite the same as for the predefinitions of the frames.	In contrary to the System Constraint Description the final System Configuration Description describes all gateways with all their gateway entries. No matter if they were predefined (System Constraint) or if they were generated by the System Generator.
`SwCompToEcuMapping` (4.1.1)	The mapping of SW Components to ECUs can be predefined. The predefinition will force the system generator to use the specified mapping. Thus, with the `SwcToEcuMapping` element it is possible to describe that one or more SW Components must be mapped to a specific ECU.	In a completed System Configuration Description, all SW components are mapped to ECUs. The mapping in the System Configuration Description is described by one `SwcToEcuMapping` element for each `ECUInstance` used in the system.
`MappingConstraint` (4.1.3) `ComponentCluster` (4.1.3.1) `ComponentSeparation` (4.1.3.2)	There may be system constraints that limit the system generators freedom to map SW components to arbitrary ECUs. These system constraints can be necessary e.g. for optimization and safety reasons to make additional guidelines for the System Generator.	After the mapping has been completed, the system configuration will contain mapping descriptions for all elements, and the mapping constraints are obsolete. But that does not mean that mapping constraints have to be deleted after the system generation step. By deleting the mapping constraints you would lose the information why a mapping of a SW Component to an ECU is chosen.

Meta-classes, Chapters (cont.)	*Usage to describe the System Contraints (cont.)*	*Usage to describe the System Configuration (cont.)*
`DataMapping` (4.2) `SenderReceiverToSignalMapping` (4.2.1.1) `SenderReceiverToSignalGroupMapping` (4.2.1.2) `ClientServerToSignalGroupMapping` (4.2.1.3)	The System Constraint Description may describe the predefined mapping of SW Components to certain ECUs (see chapter 4.1.1). Only if such a mapping exists, it is also reasonable to define the mapping of the data exchanged between those mapped SW components by a predefined mapping of data elements to the Communication Matrix.	In contrary to the System Constraint Description the final System Configuration Description contains all data mapping definitions. No matter if they were predefined (system constraint) or if they were generated by the System-Generator.
`SignalPathConstraint` (4.2.2) `CommonSignalPath` (4.2.2.1) `ForbiddenSignalPath` (4.2.2.2) `PermissibleSignalPath` (4.2.2.3) `SeparateSignalPath` (4.2.2.4)	It can be necessary e.g. for optimization and safety reasons to make additional guidelines for the System Generator, which specific way a signal between two Software Components should take in the network without defining in which frame and with which timing it is transmitted.	Signal paths are not an obligatory part of the System Configuration Description. In the final System Configuration Description every signal is assigned to a frame. Thereby the paths of the AUTOSAR-Signals are implicitly described. But that does not mean that signal path information have to be deleted after the system generation step. By deleting the signal paths you would lose the information why you have chosen e.g. a specific frame for a signal. If you extend or change the system at a later date the missing information about signal paths could lead to a not wanted signal mapping if the system Generator remaps the signals.

Table 7.1: Usage of the System Template

8 System Extract of the System Configuration Description

This chapter describes contents and creation of the AUTOSAR work product System Extract, based on Meta Model elements contained in the System Template and Software Component Template.

The System Extract is introduced to allow a collaboration between an OEM and a Supplier.[1] The OEM/Supplier Collaboration scenario is described in more detail in chapter 8.1.

The OEM is often only interested in the required functionality and the integration of the functionality into the System. Thus the OEM provides a basis for designing a subsystem, which is developed by the supplier. One difference to the ECU Extract is that the System Extract is not fully decomposed and still needs to be refined before it forms the basis for the ECU configuration. Another difference is that a System Extract is not fixed to an ECU. It is possible that the System Extract covers not only one but several ECUs.

The System Extract is using the same meta model elements as the System Configuration Description, with the difference that the `System` class refers to one specific subsystem rather than the complete System. This shall formally be marked by setting the `System's category` attribute to `SYSTEM_EXTRACT` according to table 1.2.

The same rules and constraints apply to the System Extract as for the System Description. A model transformation between a System Description and a System Extract is not necessary. In the System Extract the OEM strips all information from the System Configuration Description that is not needed for the definition of the subsystem. There is one exception to this simple "remove" rule: the communication mapping may need to be extended, which will be described in more detail in chapter 8.2.

In contrast to the ECU Extract the System Extract contains Software Compositions. It is even possible that some of the existing compositions are empty. Such empty compositions in the System Extract represent subsystems that need to be refined by a Supplier. Figure 8.1 shows an example where a System Configuration Description is stripped down to a subsystem.

[1] Collaboration scenarios between different departments of an OEM are also supported by the System Extract. For the sake of simplicity such scenarios are not addressed here.

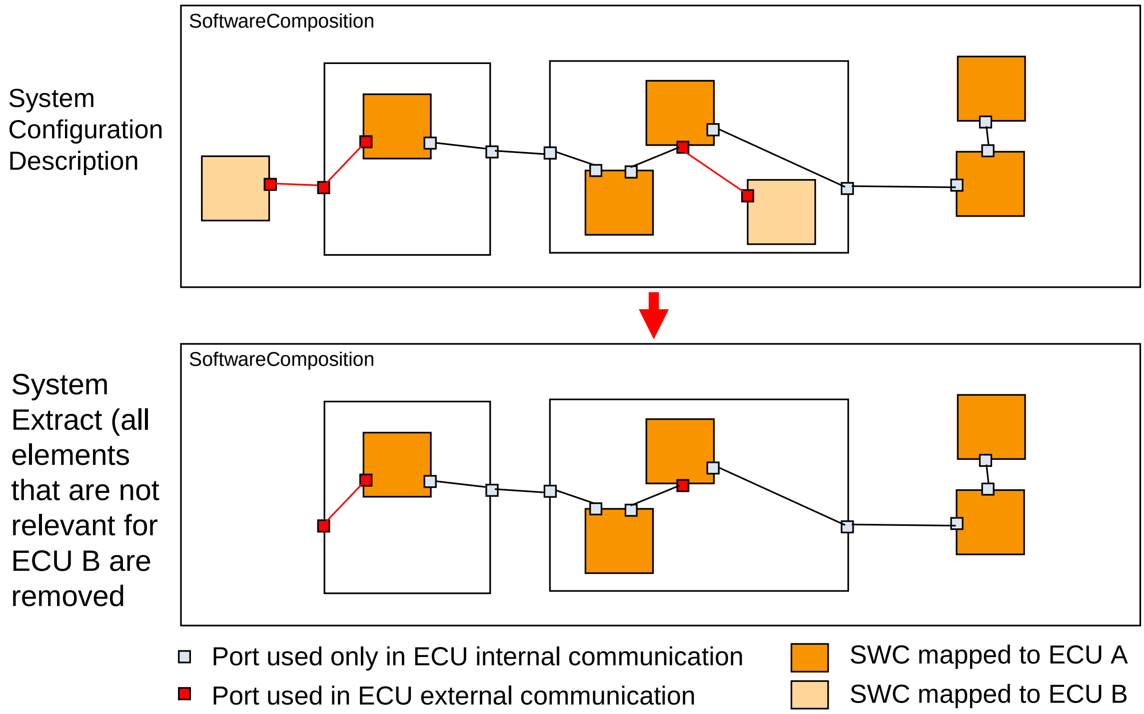

Figure 8.1: System Extract creation: irrelevant elements are removed from the System Description

8.1 OEM/Supplier Collaboration Szenario

In an important collaboration scenario, an OEM commissions a supplier to provide implementations of one or more functionalities to be integrated into an AUTOSAR system in the form of Application Components. The OEM is primarily interested in the required functionality and the interfaces defining the integration of the Software Component into the System VFB rather than the internal structure of such a component. On the other hand, the supplier, delivering both the component implementation in combination with the ECU it is destined to run on may claim the internal structure of such a higher-level component contains substantial intellectual property, and hence may not want to disclose its internal works to the OEM.

Effectively, the use case can be described in the following manner:

- The OEM generates a System Extract from the System Description. From the System Description all elements are removed that are not relevant for the design of the subsystem, such as SW components or topology elements.
- The OEM can deliver a sub-structure of Software Compositions or even Atomic Software Components in the System Extract. But the System Extract can also contain empty Software Compositions. The OEM shall have the possibility to define only the outer shell of a Software Composition that is to implement a certain functionality. Such an empty `CompositionSwComponentType` does contain all the provided and required ports with the included Com Specs describing the

requested component's outside communication needs. But it does not need to contain SwComponentPrototypes or SwConnectors at this stage.

- Such empty components are added to a System's VFB, the outside ports are connected with other components in the VFB. However, at this stage the inner structure of such CompositionSwComponentType can still be left empty.
- The System Extract contains the mapping of components to the target ECUs, including the empty compositions. Signal mappings affecting the empty compositions are targeting the CompositionSwComponentType's ports.
- The OEM delivers the System Extract to the Supplier.
- The Supplier adds the substructure to the empty CompositionSwComponentTypes by adding SwComponentPrototypes and SwConnectors. This once more leads to a hierarchical VFB, effectively the Supplier creates a local System Description for his subsystem.
- The Supplier adjusts the Signal mappings to the actual ports of the inner AtomicSwComponentType prototype.
- The Supplier generates the ECU extract from his ECU-local system description. The resulting ECU extract does not include prototypes of type CompositionSwComponentType any longer.
- Based on this ECU extract the actual ECU configuration is done

8.2 Data Mapping in the System Extract

As mentioned before, there is a slight complication to the simple "remove" rule. This can be shown best with an example.

Example: Assume a simple topology with two ECUs A and B and three PDUs X (sent from A to B), Y (sent from B to A) and Z (sent from B to A) as shown in Figure 8.2.

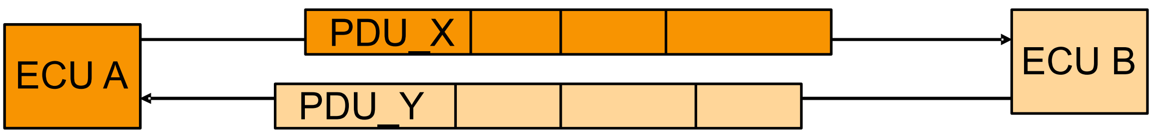

Figure 8.2: Example topology with two ECUs and two PDUs exchanged between them

Furthermore assume a SW composition as shown in Figure 8.3. It consists of six SwComponentPrototypes 'A1' to 'A3' (aggregated in composition 'SwCompA'), 'B1' / 'B2' (aggregated in composition 'SWCompB'), 'C1' (aggregated in composition 'SWCompAplusBplusC') and an empty Software Composition 'SWCompC'.

The overall composition 'SWCompAplusBplusC' aggregates 'SwCompA', 'SWCompB', the empty 'SWCompC' and the SwComponentPrototype 'C1'.

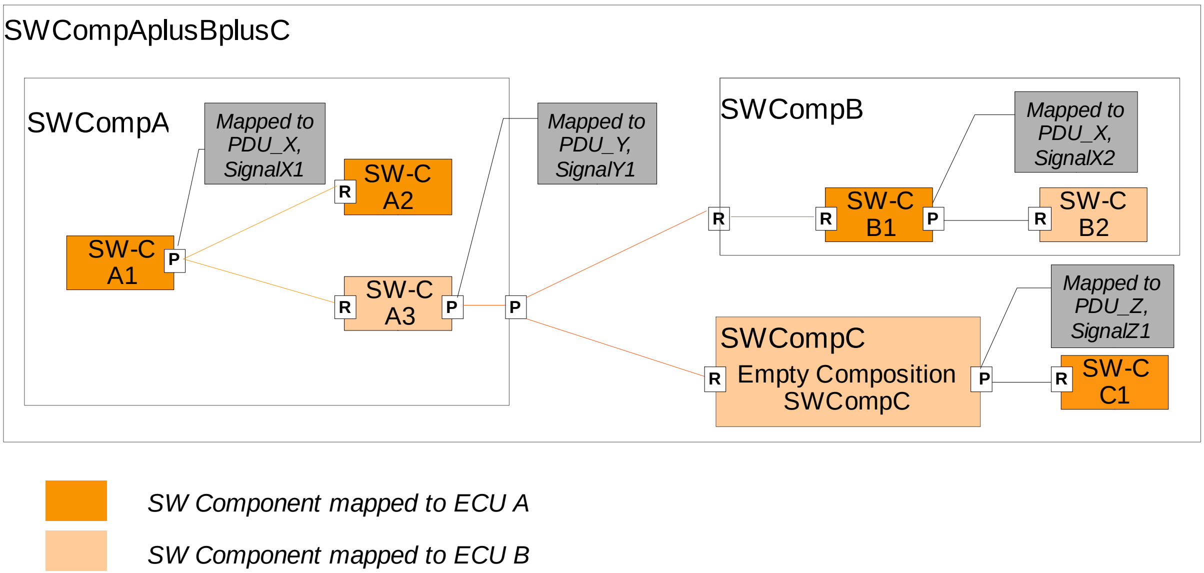

Figure 8.3: Example SW composition with mapping information

The atomic SW components ‘A1’, ‘A2’, ‘B1’ and ‘C1’ are mapped to ‘ECU A’. The atomic SW components ‘A3’, ‘B2’ and the empty Software Composition ‘SWCompC‘ are mapped to ‘ECU B’. The data sent from

- ‘A1’ to ‘A3’ is mapped to ‘PDU_X’, ‘SignalX1’,
- ‘B1’ to ‘B2’ is mapped to ‘PDU_X’, ‘SignalX2’ and
- ‘A3’ to ‘B1’ and ‘A3’ to ‘SWCompC‘ is mapped to ‘PDU_Y’, ‘SignalY1’
- ‘SWCompC‘ to ‘C1’ is mapped to ‘PDU_Z’, ‘SignalZ1’

As usual, the data mapping rules refer to the data element in the P-Port of the sending SW component. Note that data mappings can be performed on compositions and on atomic SWC as described in chapter 4.2.1. [2]

Figure 8.4 shows how the System extract for ECU A and for ECU B of this SW composition would look like: Only those elements are included that are relevant for the subsystem.

[2]Data mapping is allowed on empty compositions and on compositions that contain atomic SWCs.

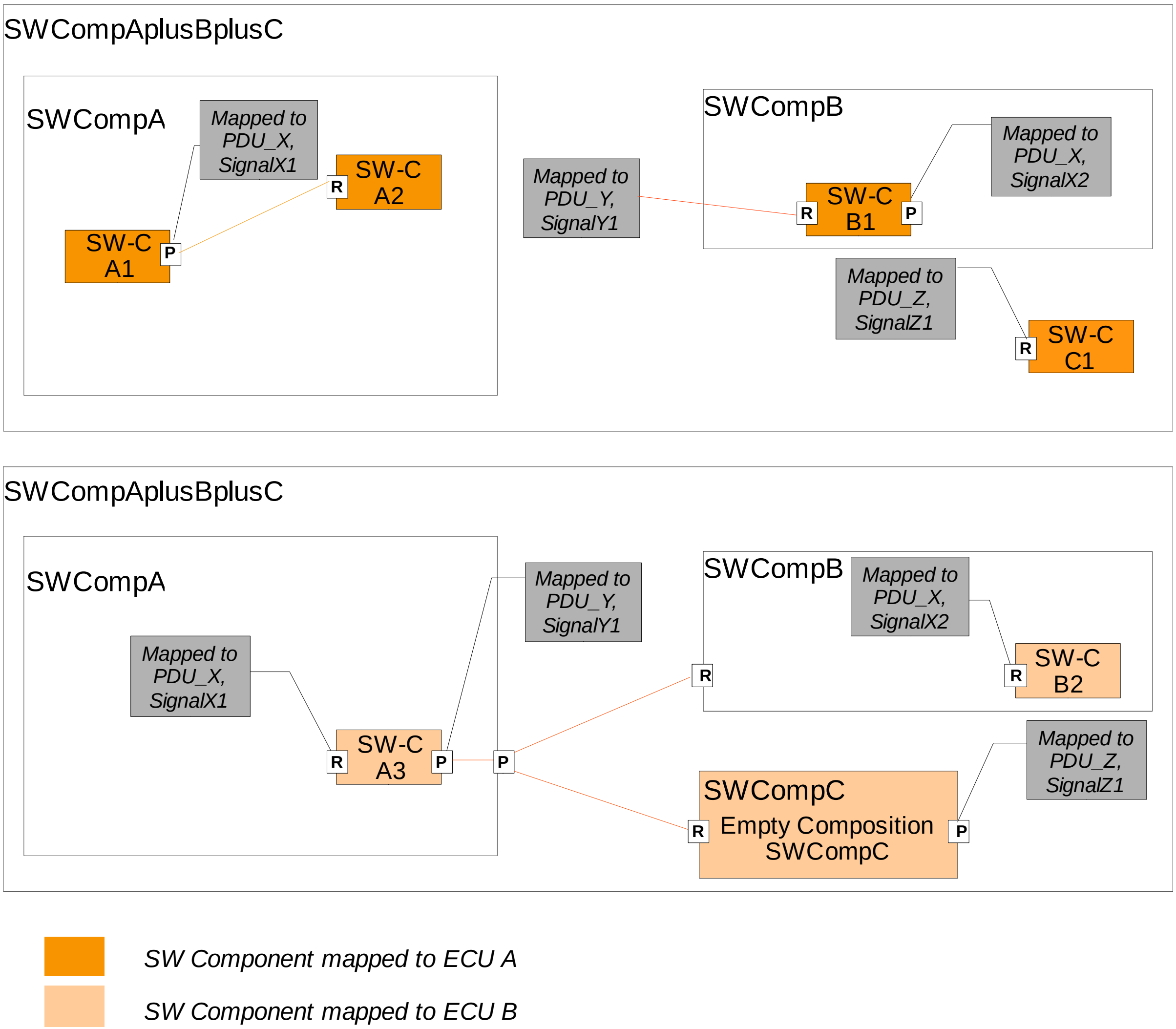

Figure 8.4: Example System extract for ECU A (upper figure) and ECU B (lower figure) of above introduced composition

In both figures all `SwComponentPrototypes` and Compositions that are mapped onto the ECU are included. The connectors between these `SwComponentPrototypes` are also included. Furthermore, the relevant topology information and communication matrix have to be included, but they are out of scope of this example.

Connectors that were used to connect to SW components that are not included in the System Extract are not included. Instead, the mapping to a signal in a PDU is used to identify the source/destination of that data.

The problem that new mapping rules have to be added arises for example in the System Extract for 'ECU A' with the mapping to 'PDU Y', 'SignalY1': Since SW component 'A3', which was referenced in the original mapping, is no longer included, the data mapping needs a new data element in a port to reference to. In the example, it is the required port of 'B1', so that the Supplier has the information that B1 receives the data via 'PDU Y'.

8.3 SW component inclusion and top level data mapping

In section 8.2 the approach is to provide the data mapping on the ports of the software components which are mapped to one ECU. Since the granularity of mapping software components to ECUs is possible for individual atomic software components this approach may result in many data mappings from different software component ports to the same system signal (depending where in the hierarchical structure they are located).

An alternative approach is to provide the complete communication information of the whole System Extract on the top level software composition and perform the data mapping on the ports of the top level software composition only. This approach is illustrated in figure 8.5.

Ports are created on the top level software composition representing the external communication of this ECU. Delegation connectors are created to establish the communication of the external software components with the software components inside the local ECU.

In figure 8.5 the software components X, Y and Z are mapped to remote ECUs. Their communication needs are collected in ports on the top level software composition and the communication is delegated via connectors inside the hierarchical software component structure.

In this example the approach for X and Y is trivial since there are only some delegation connectors required to connect the ports of the top level software composition with the ports of the respective software components.

But for software component Z the approach needs to be extended, because the communication on system level is designed to happen inside the composition V. In this case the communication needs to be delegated out of the composition (creation of delegation ports and connectors inside the composition V) to be visible in the top level software composition. Then again the approach of connection to the top level software composition can be applied.

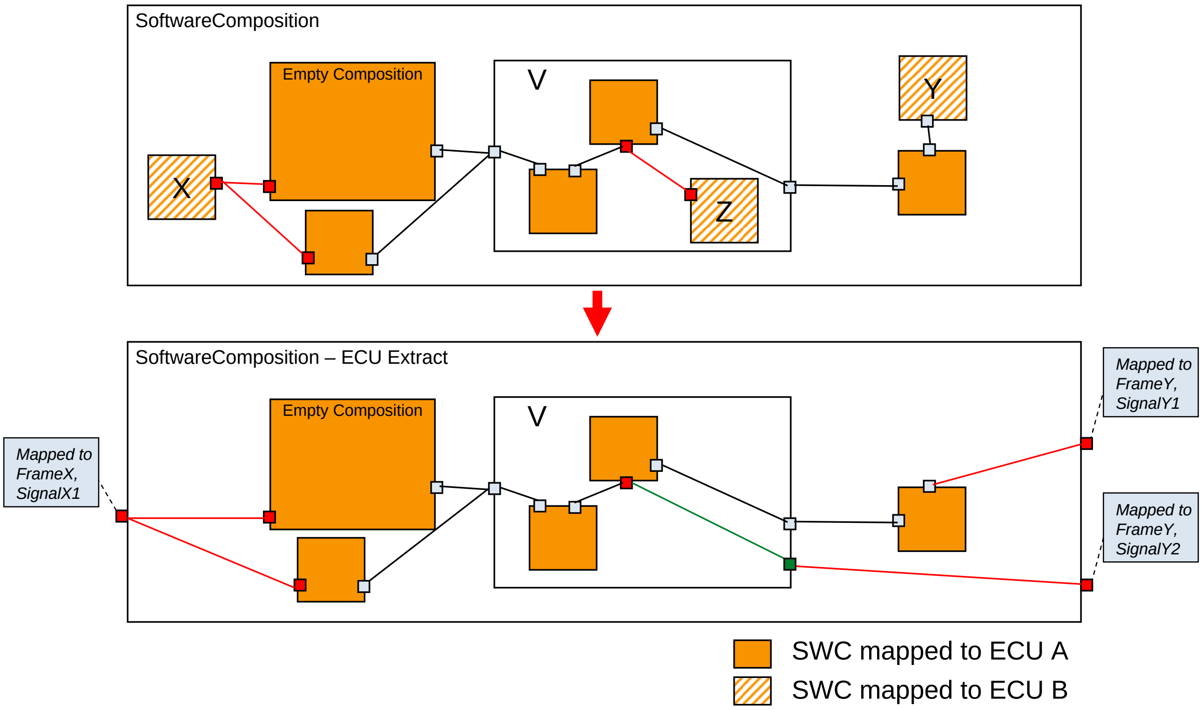

Figure 8.5: Example with software components mapped to two ECUs

9 ECU Extract of the System Configuration Description

This chapter describes contents and creation of the AUTOSAR work product ECU Extract, based on Meta Model elements contained in the System Template and Software Component Template.

The ECU Extract represents the view of one specific ECU onto the overall System Configuration Description. The ECU Extract forms the basis for configuring that particular ECU in focus.

For instance, RTE configuration fundamentally depends on the number and types of Software Components deployed onto the ECU; Services are configured according to those Software Components' service needs; the COM-stack BSW modules will be configured considering the ECU's participation in the overall System Network Topology and Communication.

The ECU Extract is using the same meta model elements as the System Configuration Description, with the difference that the `System` class refers to one specific ECU rather than the complete System. This shall formally be marked by setting the `System's category` attribute to `ECU_EXTRACT` according to table 1.2.

In order to keep ECU configuration focused and manageable despite the complexity of a full System Configuration, the ECU Extract shall only contain the subset of information relevant for configuring the targeted ECU. This comprises firstly elements that are directly deployed on the ECU, secondly elements that span across ECUs, and thirdly such that are located on a remote ECU but affect the targeted ECU's configuration. All other information shall be stripped from the System Configuration Description or from the System Extract when creating the ECU Extract.

AUTOSAR VFB Descriptions naturally form hierarchies of Software Compositions. Consequently, in the System Configuration the SWC-related information for different ECUs is not separated but in general is intermingled. In contrast, for the task of ECU configuration (RTE configuration, Service Configuration, Measurement and Calibration) a hierarchically "flat view" on the Software Components running on the ECU is preferable over a hierarchical view, which is more favored by application-software development. Thus, deriving an ECU Extract actually is a model transformation, following a set of rules described in the following sections.

As System- and ECU development typically happens in iterations, the use case of repeatedly extracting the information from an incrementally changing System Configuration needs to be considered. In particular, it must be possible to detect changes between consecutively generated ECU extracts in order to selectively update the existing ECU configuration (9.5).

AUTOSAR supports the definition and consequently the handling of Variability in the System Configuration. According to the specified binding time associated with a particular `VariationPoint`, typically some of these variants will already be resolved at the time of ECU Extract. If however the binding time occurs in a later stage of the AUTOSAR methodology, i.e. during ECU Configuration or later, the variability needs to be carried over to the ECU Extract. This also holds true for Variation points that ultimately are resolved at system configuration time but affect post-build configuration parameters. (9.6)

The ECU Extract logically forms one entity. Therefore, for ease of readability the rest of the chapter assumes just one file, "the XML file". However, it explicitly is allowed to split the ECU Extract over several files.

9.1 Topology

Only those Topology elements relevant for the ECU in scope are taken over from the System Configuration Description into the ECU Extract.

- The ECU Extract is always associated with exactly one ECU. Therefore exactly one `EcuInstance` is included in the ECU Extract along with all classes included in `EcuInstance` by composition: `CommunicationControllers` and `CommunicationConnectors` with all their `CommConnectorPorts`.
- A `CommunicationCluster` is included along with all its `PhysicalChannels` if at least one `PhysicalChannel` is used by the `EcuInstance`. In other words, if at least one of the included `CommunicationConnectors` is connected to any of a cluster's `PhysicalChannels`, the whole `CommunicationCluster` and all its `PhysicalChannels` are included.
- From the used `PhysicalChannels`, only those `FrameTriggerings`, `PduTriggerings`, `ISignalTriggerings` shall be included that are used by the ECU, e.g. they are associated with a `FramePort`, `IPduPort`, `ISignalPort` belonging to one of the ECU's `CommunicationConnectors`. *Note:* Including just a subset of a `PhysicalChannel's` triggerings is possible without changing the `PhysicalChannel`-Element itself because of the ≪`splitable`≫ stereotype applied on the `PhysicalChannel` / Triggering composition associations.

As the Topology elements are not modified when taken over into the ECU Extract, their package structure and short names are not touched (see section 9.4.1).

9.2 Top-level Software Composition

In the AUTOSAR System Configuration Description the application software composition is hierarchic by nature as described in chapter 3. When mapping Software Component Prototypes onto concrete ECUs using the `SwcToEcuMapping` class (section 4.1.1), either individual Component instances (of type `AtomicSwComponentType`), or whole component subsystems encapsulated by an `CompositionSwComponentType` are deployed onto one specified `EcuInstance`.

In contrast, the ECU Extract only contains those SwComponentPrototypes of AtomicSoftwareComponentType which are effectively mapped onto the ECU in focus. In order to obtain this ECU-centric view, the hierarchical structure of the System Configuration Description needs to be transformed into a 1-layer representation, where one distinguished CompositionSwComponentType hosts all instances of atomic software components to run on the ECU. More precisely, this ECU-level RootSwCompositionPrototype contains one SwComponentPrototype for each instance of any AtomicSwComponentType which has been mapped onto the ECU. In the ECU Extract the resulting software composition is a flat structure where the included SwComponentPrototypes become real SWC instances, reflecting the actual resource needs on the targeted ECU.

The transformation from hierarchical to flat Software Component structure includes a number of steps, to be performed per ECU. The list below outlining this process assumes that the extraction is done for the first time; if an ECU Extract already exists from a previous development cycle, the extract shall merely be updated instead of created; for more details on iterative development see section 9.5.

- Create the one CompositionSwComponentType which will represent the ECU's SW subsystem (in further steps referred to as ECU flat view)
- To this ECU flat view, add a SwComponentPrototype for each instance of any AtomicSwComponentType mapped onto the ECU. Copy all the identifiable information from the originating SwComponentPrototype, but assign an unique short name to the new element. The newly created SwComponentPrototypes are typed by the original AtomicSwComponentType.
- Unroll the connector paths leading to and from the included components:
  - For ECU internal communication, use AssemblySwConnector to connect ports.
  - For ECU external communication, add delegated Ports to the ECU flat view CompositionSwComponentType. Each delegated Port shall be connected via a DelegationSwConnector with ports of the included components that are used for the external communication.
  - VariableDataPrototypes and ClientServerOperations of the DelegatedPorts are mapped to System Signals.
- If the System Configuration Description prescribes an atomic Software Component prototype an Implementation by using SwcToImplMapping, a corresponding constraint needs to be created in the ECU Extract of the targeted ECU. The SwcToImplMapping's component instance reference needs to be adjusted to the flat representation, while maintaining the original reference to the Implementation.

Figure 9.1 illustrates the process of flattening the hierarchical Software Composition into an ECU Flat View representation, as outlined in the previous paragraphs. The following sections explain the concrete transformation steps in more detail.

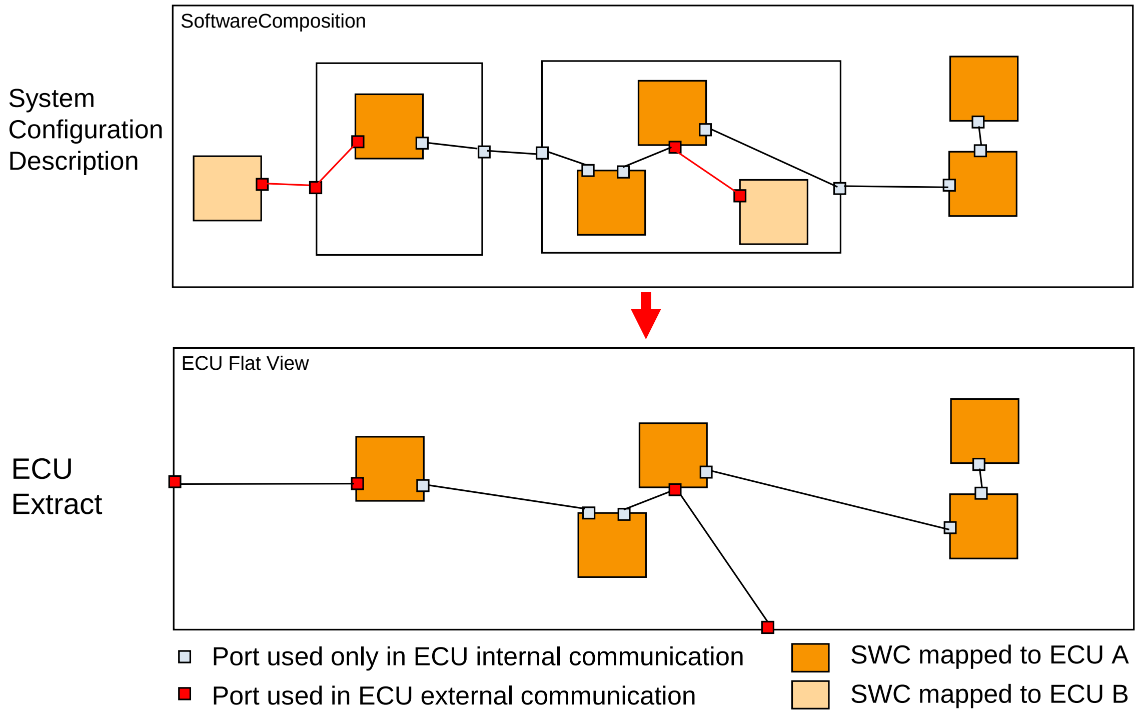

Figure 9.1: Flattening of a hierarchic Software Composition into an ECU Flat View, and the distinction between ports used in internal and those used in external communication.

9.2.1 ECU Flat view

The first step of extracting the ECU specific Software View is the creation of a new `CompositionSwComponentType` (further referred to as ECU flat view). This new element serves as a container for collecting prototype instances of all atomic application software components deployed on the ECU. In order to include the ECU flat view into the actual ECU Extract work product, the `System` representing the ECU Extract must have its child class `RootSwCompositionPrototype` pointing to this ECU flat view.

Next, all `SwCompToEcuMappings` present in the System Configuration Description need to be analyzed according to the precedence rules (Section 4.1.1) in order to establish the exact set of `AtomicSwComponentType` instances to be included on this ECU.

For each of these component instances, regardless of their order of depth in the System Configuration Description's Component hierarchy, exactly one `SwComponentPrototype` shall be created in the ECU flat view `CompositionSwComponentType`. The new element's description and type information shall be taken over from the original `SwComponentPrototype` as present in the System Configuration Description. As an important exception to this rule, the `SwComponentPrototype's shortName` must be unique in the name space formed by the ECU flat view.

The special case of prototypes of ParameterSwComponentTypes and ServiceProxySwComponentTypes is treated in almost the same way. The only difference is that these component types can be instantiated at most once per ECU and that for a given prototype in the system, instances on several ECUs can be created. Since ParameterSwComponentTypes do not have connectors across ECU boundaries, and ServiceProxySwComponentTypes can only receive but not send signals over the network, the replication of theses components on several ECUs does require any special treatment of their communication properties.

9.2.2 Internal Communication

When flattening the Software Composition for the ECU Extract, not only all of the ECU's Software Components are to be collected in the ECU flat view, but also any connection existing between ports of the included SwComponentPrototypes needs to be projected onto the same RootSwCompositionPrototype.

In the hierarchical software composition, communication between Software Components is specified by a combination of AssemblySwConnectors and DelegationSwConnectors. Several DelegationSwConnectors may be combined in case of a multiple-level delegation, however there will always be exactly one AssemblySwConnector on the outermost CompositionSwComponentType the port is delegated to.

In the ECU flat view, any such number of stringed together SwConnectors effectively connecting two ports of Software Component Instances mapped to the same ECU are resolved to exactly one AssemblySwConnector per connected port pair. As there are no additional levels of “inner SwComponentPrototypes”. DelegationSwConnectors are only used to display the outside communication of an ECU in the ECU flat view.

[constr_3019] In the flat ECU extract each required interface must be satisfied by connected provided interfaces ⌈ In case of the flat ECU extract all dataElements specified by the SenderReceiverInterface of the required port need to be supplied by some of the provided ports being connected with SwConnectors. ⌋

For the System Configuration Description, the Software Component Template Specification [4] allows a CompositionSwComponentType's outer PortPrototype to be connected to more than one inner port, observing a set of compatibility rules between the outer and the inner port's SenderReceiverInterfaces. Such a “merge” and “split” functionality for mixing VariableDataPrototype is used to limit the number of SwConnectors required to connect Ports on higher VFB levels and thus reduce complexity in the wiring of such higher-level CopmpositionComponentTypes. On the other hand this means that an AssemblySwConnector in a hierarchical VFB may expand to more than one Port-Port pair. Naturally, in the ECU flat view such “hidden” additional connections need to be made explicit by unrolling them into concrete AssemblySwConnectors. The following paragraph suggests a way how such an unrolling of connectors may be accomplished.

Starting with the top-level RootSwCompositionPrototype indicating the outermost CompositionSwComponentType, the hierarchical software model of SwComponentPrototypes is recursively iterated; for each prototype of CompositionSwComponentType, all its AssemblySwConnectors are being iterated. For each such found AssemblySwConnector both connector ends are evaluated for DelegationSwConnectors further delegating the connection: In order to consider the use cases of signal “merge” and “split”, all possible communication partners need to be identified, recursively following DelegationSwConnectors in both directions. For each identified pair of ProvidedPort and RequiredPort *actually exchanging* Information one AssemblySwConnector will be created in the ECU flat view.

The following rules must be followed when PortInterfaceElement Mappings are converted for the flat view. PortInterfaceMappings supports the connection of Ports typed by two different PortInterfaces with unequal named PortInterface elements. More details can be found in [4].

- When unrolling a string of connectors into a single connector all compatibility rules and PortInterfaceMappings of the individual connectors need to be considered for determining which data elements are being transferred between provider and requester. If data elements are to be filtered out a portInterfaceMapping shall be provided to the flatten connector such that only the transfered data elements are included in the mapping.
- When unrolling a string of connectors into a single connector all of the PortInterfaceMappings of the individual connectors need to be considered for combining them into a single PortInterfaceMapping to be associated with a new connector.

9.2.3 External Communication

In a complete System Configuration Description, whenever two SwComponentPrototypes are specified to communicate across EcuInstances, the details of this communication need to be fully specified: VariableDataPrototypes of SenderReceiverInterfaces and ClientServerOperations of ClientServerInterfaces are mapped onto SystemSignals as carriers of information transported across the network. According to 4.2, each instance of a AutosarDataPrototype that is to be sent over AUTOSAR COM shall be mapped exactly once onto its individual SystemSignal, regardless of how many components receive the information or over how many busses the Signal is transported.

As described above, deriving the ECU Extract from System Configuration Description or from a System Extract means that all SwComponentPrototypes to be included in the extract are recreated in an ECU flat view. Consequently, each DataMapping concerning a component instance to be mapped onto the ECU requires that a corresponding DataMapping be created in the Ecu extract.

The ECU flat view contains delegated ports to display the outside communication of an ECU. VariableDataPrototypes and ClientServerOperations of these delegated ports are mapped to System Signals. The original instance references indicating the mapped AutosarDataPrototype need to be adjusted to the new "flat" location in the ECU flat view.

While for the System Configuration Description it is sufficient to describe DataMappings only on the provider side, the ECU extract additionally requires such DataMappings on the requiring side's ports. In this case, a new DataMapping maps to the existing SystemSignal, previously defined in the System Configuration Description on the provider side. This is explained in more detail in the following example, that is a continuation of the example from figure 8.3 in chapter 8.2:

Figure 9.2 shows how the ECU extract for ECU A of the SW composition that is defined in figure 8.3 would look like: Only those atomic SW components are included that are mapped to ECU A. The hierarchy present in the System Configuration Description has been flattened into CompositionSwComponentType 'EcuAFlatView', including newly created SwComponentPrototype 'A1E', 'A2E', 'B1E' and 'C1E' for the component instances mapped to ECU A.

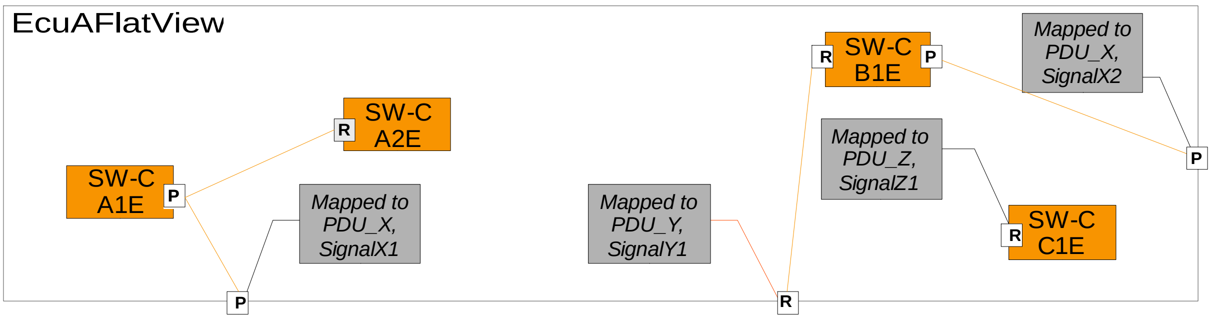

Figure 9.2: Example ECU extract for ECU A of above introduced composition

The connectors to the outside ports (ECUFlatView composition ports) and connectors that represent intra-ECU communication (in our example, only 'A1E' to 'A2E') are included. The VariableDataPrototypes and ClientServerOperations in the outside ports are mapped to SystemSignals. This data mapping and the communication description is used to identify the source/destination of that data.

Furthermore, the relevant topology information and communication matrix have to be included, but they are out of scope of this example.

The problem that new mapping rules have to be added arises with the mapping to 'PDU_Y', 'SignalY1': Since SW component 'A3', which was referenced in the original mapping, is no longer included, the data mapping needs a new data element in a port to reference to. In the example, the data of the required port of 'B1E' is referenced, so that the ECU generator has the information that 'B1E' receives the data via 'PDU_Y'.

9.2.4 Port Groups

A SwComponentType can optionally define PortGroups which allow to group ports according to logical criteria, e.g. according to shared communication resources (see [4]). A PortGroup of a CompositionSwComponentType can be linked to "inner" PortGroups of the aggregated component prototypes. Since the main purpose of this grouping is to configure the behavior of mode managers on an ECU, this information must be preserved and broken down into the ECU extract.

The resulting CompositionSwComponentType in the ECU flat view will contain a set of PortGroups which refer to the linked inner port groups of the atomic component prototypes. To get to this result, the following steps must be applied in the extraction process:

- Recursively ignore all port groups in CompositionSwComponentTypes in the hierarchical structure, which are not linked to any inner groups to be mapped on this ECU.
- In the remaining structure of linked port groups find out the top level groups (i.e. which are not referred by any higher level group on this ECU) and put an element representing each top level group into the CompositionSwComponentType of the ECU flat view. This can result in name conflicts, which should be resolved by a suitable algorithm.
- Link these top level groups to the inner groups of the atomic component instances of the flat view according to the links found in the hierarchical structure. Naturally, the top level port groups in the ECU flat view are not directly referring any ports and due to the first step they should be linked to at least one inner group.
- The port groups in the atomic software components on the ECU should be unchanged.

9.3 Communication

In explaining how SystemSignals are handled in the ECU Extract, Section 9.2.3 touched on the topic of inter-ECU Communication. However, in order to enable the ECU Configuration of the COM-Stack, the relevant information of all layers of the AUTOSAR COM-Stack needs to be present in the ECU Extract, including the central Communication classes ISignal, PDU and Frame.

The above mentioned Communication elements have dependencies on each other, for ordinary COM-communication this means:

- Frames are assembled from one or more PDUs.
- ISignalIPDUs carry their information in form of ISignals.
- ISignals as interaction points between RTE and COM refer to SystemSignals.

Note that the above list is not complete; TP and NM require additional elements. However, for the sake of clarity the following paragraphs describes the standard use case of a direct Signal-based communication between two ECUs. Once the handling of this case is understood, the additional model elements as NPdu, NmPdu, SystemSignalGroup etc. can be handled following the same basic principles.

For the ECU Extract only the ECU-relevant subset of information present in the system-wide communication is to be considered. In order to establish this set of information, the dependencies in the list above are being followed.

9.3.1 Frame

In a complete System Configuration Description, every outside communication of an ECU will either be associated with an outgoing or and incoming frame. The exact number and types of frames to be received or sent by an ECU is determined by the Communication Matrix (Chapter 5).

According to the selection rules for the Topology (9.1), the ECU Extract contains all FrameTriggerings associated with Frames that are of any interest to the EcuInstance: If a particular FrameTriggering refers to a FramePort of type 'out' the associated Frame is to be sent by the ECU, if it refers to an 'in' port the Frame is to be received. Therefore, the following selection rule applies:

- The ECU Extract shall contain all Frame elements which are referenced by any included FrameTriggering.

9.3.2 PDU

Frames are assembled from one or more PDUs. In order to include all required PDU elements, the following selection criteria apply:

- The ECU Extract shall contain all PDU elements which are referenced by any included Frame's PduToFrameMapping.
- The ECU Extract shall contain all PDU elements which are referenced by any included PduTriggering.
- For multiplexed PDUs, additionally all SignalIPdus referenced by the MultiplexedIPdu's static and dynamic parts need to be included.

The second criterion is e.g. required in a pure post-build configuration scenario, where the frame-layout may not be completed at the time of ECU Extract creation.

— AUTOSAR CONFIDENTIAL —

9.3.3 ISignals and ISignalGroups

ISignalIPdus carry their information in form of ISignals or ISignalGroups. In order to include all required ISignal and ISignalGroup elements, the following selection criteria apply:

- The ECU Extract shall contain ISignal elements which are referenced by included ISignalIPdu's ISignalToIPduMapping. One exception are Pdu Gateways. Signal definitions that are not directly relevant for gateway ECUs in case that the Pdu is routed as a whole (Pdu Routing) shall be omitted. See Section 9.3.5 for more details.
- The ECU Extract shall contain all ISignal elements which are referenced by any included ISignalTriggering.
- The ECU Extract shall contain ISignalGroup elements which are referenced by included ISignalIPdu's ISignalToIPduMapping. One exception are Pdu Gateways. Signal Group definitions that are not directly relevant for gateway ECUs in case that the Pdu is routed as a whole (Pdu Routing) shall be omitted. See Section 9.3.5 for more details.
- The ECU Extract shall contain all ISignalGroup elements which are referenced by any included ISignalTriggering.

Like in the case of the PDU inclusion rules, the second and fourth criterion is required in scenarios with incomplete PDU modeling due to post-build configurability of the communication matrix.

9.3.4 SystemSignal and SystemSignalGroup

Whereas the rules specified in Section 9.2.3 for the inclusion of SystemSignal comprise all SystemSignals that are being used by the Software Components in the ECU, the inclusion rules above stated for SignalIPdus and ISignals may require the inclusion of additional SystemSignals. Also, strictly speaking both SystemSignals and SystemSignalGroup need to be considered. The complete inclusion rules for AbstractSignals are:

- The ECU Extract shall contain all AbstractSignal (SystemSignal and SystemSignalGroup) elements which are referenced by any included DataMapping.
- The ECU Extract shall contain all AbstractSignal elements which are referenced by any included ISignal.

9.3.5 Gateways

Gateways that are referenced by the EcuInstance shall be included in the ECU extract. The complete inclusion rules for Gateways are:

- The ECU Extract shall contain all FrameMapping elements that are aggregated by the Gateway element.
- The ECU Extract shall contain all IPduMapping elements that are aggregated by the Gateway element.
- The ECU Extract shall contain all ISignalMapping elements that are aggregated by the Gateway element.
- Signal definitions that are not directly relevant for the gateway ECU in case that the Pdu containing these signals is routed as a whole (Pdu Routing) shall be omitted.
- Signal Group definitions that are not directly relevant for the gateway ECU in case that the Pdu containing these signal groups is routed as a whole (Pdu Routing) shall be omitted.

9.3.6 TP configuration

The TP-configuration element TpConfig and all its associated elements shall be included into the ECU Extract if the EcuInstance has an TpAddress configured in this TpConfig.

9.3.7 NM configuration

The NM-configuration element NmConfig and all its associated elements shall be included into the ECU Extract if the EcuInstance is included in the NmConfig's configured list of NmEcus.

9.4 Naming Issues

9.4.1 Package Structure

As detailed in the sections above, extracting information from the System Configuration Description into an ECU Extract is a non-trivial transformation: While some of the model elements are simply copied verbatim into the ECU Extract, it is additionally necessary to create new elements reducing parts of system-wide structures, most noticeably in flattening of the hierarchical VFB view to the ECU Flat View.

All such elements being created or modified in the process of generating the ECU extract shall reside in the same ARPackage. In order to avoid namespace conflicts with existing elements, the package shall exclusively be used for this purpose.

By creating derivation elements from elements originally contained in the System Description's package structure, duplications of names may occur. This kind of name clashes shall be resolved by a suitable naming algorithm (see section 9.4.3).

All Elements that are taken over from the System Description unchanged (e.g. AtomicSwComponentType, PortInterface, ApplicationDataType, EcuInstance, CommunicationCluster) shall remain in their original packages.

ARElements not used in the ECU extract shall not be copied to the ECU Extract XML file.

In more detail, ARPackages taken over from System Configuration Description will not be altered by the ECU extraction process, except that some ARElements will not be included in the actual XML file of the extract: ARElements which exist in the System Configuration Description but have been stripped for the ECU Extract are not actually deleted from their ARPackage, but merely are skipped in the XMLfile forming the extract. Note that having such a partial view on an ARPackage doesn't break the original ARPackage definition because the composition of PackageableElement, responsible for adding ARElements to ARPackage, is stereotyped ≪splitable≫; this means several XML files can contribute to an ARPackage, or in case of the ECU Extract an AUTOSAR description file may contain only a subset of the complete ARPackage.

9.4.2 Naming of Measurement and Calibration Data

The software component descriptions provide several means to declare data prototypes which have to be available for measurement and calibration (MCD) tools on the ECU. Together with the ECU extract it is required to provide a list of references to the description of these data for further processing in the scope of the ECU. In addition, the MCD tools need a unique name for each instance of such a data prototype. Since the data descriptions are part of the nested composition structure and are contained in reusable types (components or port interfaces), the system description itself does in general not provide unique names for those.

This means, providing such a list with references and unique names for MCD data is also a task of the ECU extractor tool. This list is part of the artifact ECU Flat Map, which is further explained below.

9.4.3 Naming of Derived Elements

When performing the extract process, name clashes may occur, necessitating a naming scheme for elements derived in ECU generation: By flattening the Software Composition Hierarchy all component instances present on the considered ECU are put in one ECU-wide software composition. Name clashes may occur for the following reasons:

1. SwComponentPrototypes taken from different Software Compositions are allowed to have identical short names in the hierarchical structure. As all SwComponentPrototypes will be located in the same ECU Flat View, the original name spaces separation no longer exists.
2. Multiple instances of the same CompositionSwComponentType are mapped to an ECU: In this case, duplicates of all contained SwComponentPrototypes will be placed next to each other in the ECU flat composition.
3. The two mechanisms just mentioned may also lead to name clashes in AutosarDataPrototypes if their names shall be used as MCD data names. In addition, reuse of a PortInterface can also lead to name clashes if it provides data elements to be used by MCD.
4. The setup of PortGroups in the ECU flat view can result in name clashes, because two port groups originating from different component types (i.e. different name spaces) may be aggregated within the flat view.

Therefore the ECU Extract generator shall take care that all elements derived or created during the extraction process have unique short names. These unique names shall be created in an initial step of the extraction process which leads to the creation of an initial ECU Flat Map. Some ways to satisfy this requirement may be:

- Use globally unique identifiers (GUID) for generating short names.
- Add a number to the original name; if done consistently the flat map approach makes this reproducable.
- Expand the name recursively by the names of the containing elements (e.g. compositions) until is is unique.
- Allow human interaction (this may be combined with an initially proposed name expansion).

The creation of a new short name is compulsory only if otherwise a clash would occur.

9.4.4 Re-use of short names assigned in previous iterations

As described in the previous section, potential name clashes during ECU extraction must be avoided by assigning unique names to the elements specifically created for the ECU Extract and for the list of MCD data per ECU. Considering the use case of iterative development (also see Section 9.5), the same names shall be assigned to existing elements in consecutive iterations. Elements which have been modified or newly introduced between two ECU extract iterations shall not use an existing short name. Additionally, the ECU extractor tool shall not re-use any short name used in any iteration from previous development phases if the meaning of the element is not exactly the same (i.e. the element's back reference into the System Configuration Description is not the same.)

9.5 ECU Extract in subsequent Cycles of Iterative Development

9.5.1 Traceability of model elements created in ECU Extract

For development scenarios in real life projects iterative development must be supported.

The following use case shall be considered:

Changes in the System Configuration Description require the recreation of an ECU extract. In the successive re-run of ECU configuration, ECU configuration parameters which were configured based on the previous ECU Extract need to be maintained for those parts in the ECU Extract that didn't change between iterations.

Consequently, there are two requirements on the extraction process:

- Elements that are present in both versions of the System Configuration Description must not change their short names between the two ECU Extracts either.
- If changes between the two versions of the System Configuration Description lead to the creation of new model elements in the ECU Extract, then these newly created elements must have new names that have not been used in previous iterations of the ECU Extract. (See also Section 9.4.4).

In order to fulfill these requirements, a back-tracing of the relevant model elements in the ECU Extract to their counterparts in the System Description shall be established. Based on these back references, short names shall consistently be re-used in iterations. Relevant elements are all those which potentially have been modified in the extraction process.

All back-tracing references are collected in one central table per ECU Extract, based on the meta-class `FlatMap`. This table collects "instance" entries for each ECU Extract element that is being created in the ECU extract transformation and for each MCD data object that has to be available in the ECU. These entries are called `FlatInstanceDescriptor`.

Each mapping entry owns two references per mapped element, one reference pointing to the target element in the ECU extract, the other one pointing to the origin in the System Description. Both of these references are deep "instance" references, requiring a tuple of context/target description.

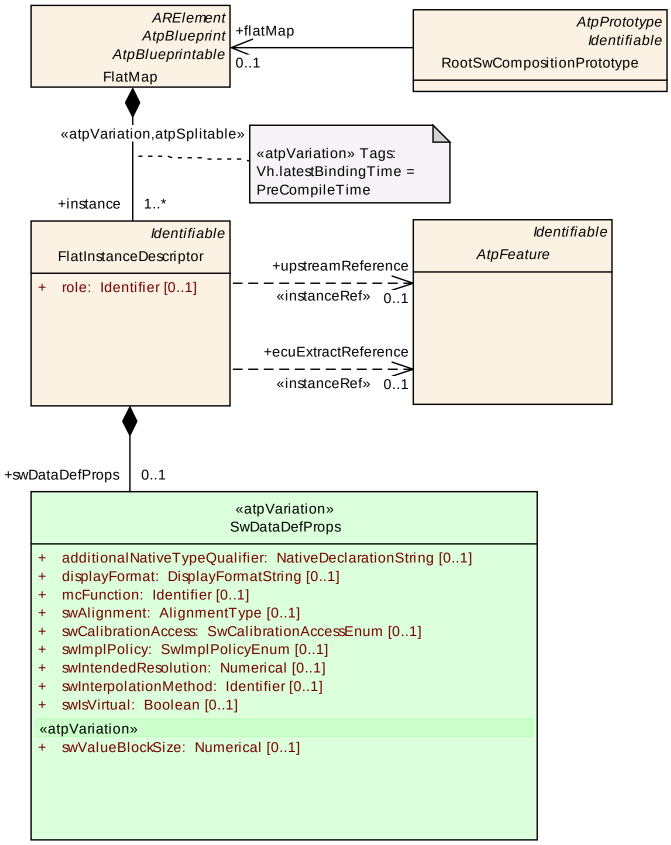

Figure 9.3: Flat Map (CommonStrucure: FlatMap)

Class	**FlatMap**			
Package	M2::AUTOSARTemplates::CommonStructure::FlatMap			
Note	Contains a flat list of references to software objects. This list is used to identify instances and to resolve name conflicts. The scope is given by the RootSwCompositionPrototype for which it is used, i.e. it can be applied to a system, system extract or ECU-extract. An instance of FlatMap may also be used in a preliminary context, e.g. in the scope of a software component before integration into a system. In this case it is not referred by a RootSwCompositionPrototype. **Tags:** atp.recommendedPackage=FlatMaps			
Base	ARElement,ARObject,AtpBlueprint,AtpBlueprintable,Collectable Element,Identifiable,MultilanguageReferrable,PackageableElement,Referrable			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***

Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
instance	FlatInstanceDescriptor	1..*	aggr	A descriptor instance aggregated in the flat map. The variation point accounts for the fact, that the system in scope can be subject to variability, and thus the existence of some instances is variable. The aggregation has been made splitable because the content might be contributed by different stakeholders at different times in the workflow. Plus, the overall size might be so big that eventually it becomes more manageable if it is distributed over several files. **Stereotypes:** atpSplitable; atpVariation **Tags:** Vh.latestBindingTime=PreCompileTime atp.Splitkey=shortName, variationPoint.shortLabel

Table 9.1: FlatMap

Class	**FlatInstanceDescriptor**			
Package	M2::AUTOSARTemplates::CommonStructure::FlatMap			
Note	Represents exactly one node (e.g. a component instance or data element) of the instance tree of a software system. The purpose of this element is to map the various nested representations of this instance to a flat representation and assign a unique name (shortName) to it. Use cases: • Specify unique names of measurable data to be used by MCD tools • Specify unique names of calibration data to be used by MCD tool • Specify a unique name for an instance of a component prototype in the ECU extract of the system description Note that in addition it is possible to assign alias names via AliasNameAssignment.			
Base	ARObject,Identifiable,MultilanguageReferrable,Referrable			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***

Attribute	*Datatype*	*Mul.*	*Kind*	*Note*
ecuExtract Reference	AtpFeature	0..1	iref	Refers to the instance in the ECU extract. This is valid only, if the FlatMap is used in the context of an ECU extract. The reference must be such, that it uniquely defines the object instance. For example, if a data prototype is declared as a role within an SwcInternalBehavior, it is not enough to state the SwcInternalBehavior as context and the aggregated data prototype as target. In addition, the reference must also include the complete path identifying instance of the component prototype and the AtomicSoftwareComponentType, which is refered by the particular SwcInternalBehavior. **Tags:** xml.sequenceOffset=40
role	Identifier	0..1	ref	The role denotes the particular role of the downstream memory location described by this FlatInstanceDescriptor. It applies to use case where one upstream object results in multiple downstream objects, e.g. ModeDeclarationGroupPrototypes which are measurable. In this case the RTE will provide locations for current mode, previous mode and next mode.
swDataDef Props	SwDataDefProp s	0..1	aggr	The properties of this FlatInstanceDescriptor.
upstreamR eference	AtpFeature	0..1	iref	Refers to the instance in the context of an "upstream" descriptions, wich could be the system or system extract description, the basic software module description or (if a flat map is used in preliminary context) a description of an atomic component or composition. This reference is optional in case the flat map is used in ECU context. The reference must be such, that it uniquely defines the object instance in the given context. For example, if a data prototype is declared as a role within an SwcInternalBehavior, it is not enough to state the SwcInternalBehavior as context and the aggregated data prototype as target. In addition, the reference must also include the complete path identifying the instance of the component prototype that contains the particular instance of SwcInternalBehavior. **Tags:** xml.sequenceOffset=20

Table 9.2: FlatInstanceDescriptor

— AUTOSAR CONFIDENTIAL —

[TPS_SYST_1000] `FlatInstanceDescriptor` roles ⌈ If a `ModeDeclarationGroupPrototype` is measurable the `FlatMap` shall contain three entries where the particular roles are set to

- CURRENT_MODE specifies the `FlatInstanceDescriptor` applicable for current mode value of the `ModeDeclarationGroupPrototype`
- PREVIOUS_MODE specifies the `FlatInstanceDescriptor` applicable for previous mode value of the `ModeDeclarationGroupPrototype`
- NEXT_MODE specifies the `FlatInstanceDescriptor` applicable for next mode value of the `ModeDeclarationGroupPrototype`

Please note that these entries may exist in a `FlatMap` even if the `ModeDeclarationGroupPrototype` is not measurable. ⌋*(SYSCT0003, SYSCT0027)*

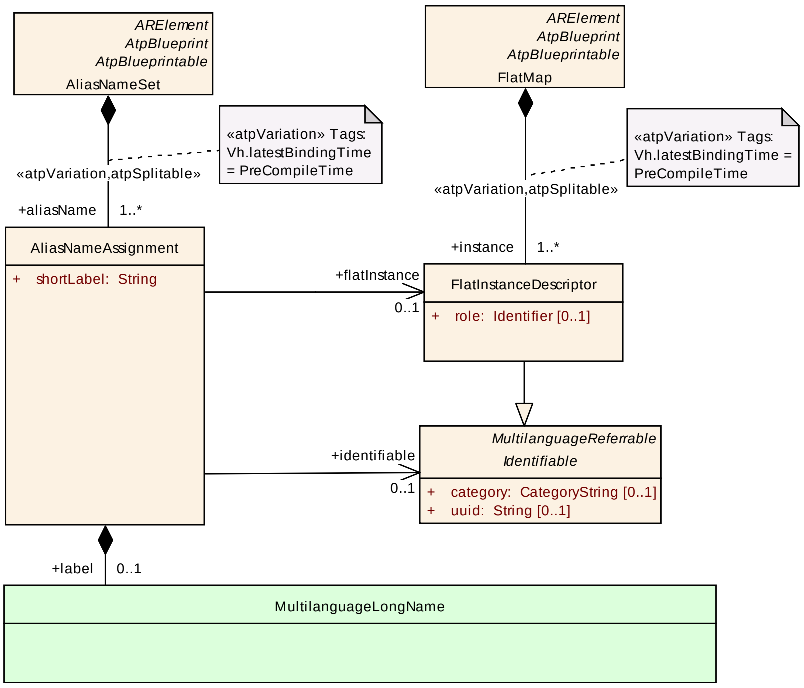

Figure 9.4: Alias Name Assignment (CommonStrucure: AliasNameAssignment)

Class	**AliasNameSet**			
Package	M2::AUTOSARTemplates::CommonStructure::FlatMap			
Note	This meta-class represents a set of AliasNames. The AliasNameSet can for example be an input to the A2L-Generator. It shall not be used by the RTE generator to generate the MC-Support. In a given instance of AliasNameSet in the bound system there must be at most one aliasName per FlatInstanceDescriptor. **Tags:** atp.recommendedPackage=AliasNameSets			
Base	ARElement,ARObject,AtpBlueprint,AtpBlueprintable,CollectableElement,Identifiable,MultilanguageReferrable,PackageableElement,Referrable			
Attribute	*Datatype*	*Mul.*	*Kind*	*Note*

Attribute	Datatype	Mul.	Kind	Note
aliasName	AliasNameAssignment	1..*	aggr	AliasNames contained in the AliasNameSet. **Stereotypes:** atpSplitable; atpVariation **Tags:** Vh.latestBindingTime=PreCompileTime atp.Splitkey=shortLabel

Table 9.3: AliasNameSet

Class	**AliasNameAssignment**			
Package	M2::AUTOSARTemplates::CommonStructure::FlatMap			
Note	This meta-class represents the ability to associate an alternative name to a flat representations or an Identifiable. The usage of this name is defined outside of AUTOSAR. For example this name can be used by MCD tools or as a name for component instances in the ECU extract. Note that flatInstance and identifiable are mutually exclusive.			
Base	ARObject			
Attribute	***Datatype***	***Mul.***	***Kind***	***Note***
flatInstance	FlatInstanceDescriptor	0..1	ref	Assignment of a unique name to a flat representation. **Tags:** xml.sequenceOffset=60
identifiable	Identifiable	0..1	ref	Assignment of a unique name to an Identifiable. **Tags:** xml.sequenceOffset=50
label	MultilanguageLongName	0..1	aggr	This represents an "Alias LongName". **Tags:** xml.sequenceOffset=20
shortLabel	String	1	attr	This attribute represents the alias name. It is modeled as string because the alias name is used outside of AUTOSAR and therefore no naming conventions can be applied within AUTOSAR. **Tags:** xml.sequenceOffset=10

Table 9.4: AliasNameAssignment

During the ECU extraction process, the ECU Flat Map will be processed in the following steps:

1. Create the entries `shortName` and `upstreamObject` of the ECU Flat Map or, if a previous version exists, try to reuse them. Resolve name conflicts.
2. Generate the ECU Software Composition.
3. Create the entries `ecuExtractObject` of the ECU Flat Map.

More details are define be the AUTOSAR methodology, see [3]. The methodology also allows to have a Flat Map for the whole system . This System Flat Map can be created and maintained independently from the ECU extraction process, but can be used as an input for the creation of the ECU Flat Map.

9.5.2 Mapping of AUTOSAR attributes to ASAM ASAP2

With the MC Support information AUTOSAR builds a bridge to tools processing ASAM ASAP2 files. In order to support the interoperability of converter tools the following mapping of AUTOSAR attributes to ASAM ASAP2 [21] (also known as "A2l" respectively "ASAM MCD 2MC") is recommended:

- If the `FlatInstanceDescriptor` references `DataPrototypes`:

  FlatInstanceDescriptor.shortName ->
  MEASUREMENT Name
  CHARACTERISTIC Name

  FlatInstanceDescriptor.(longName + desc |upstreamReference.desc) ->
  MEASUREMENT LongIdentifier
  CHARACTERISTIC LongIdentifier

  AliasNameAssignment.shortLabel ->
  MEASUREMENT [-> DISPLAY_IDENTIFIER]
  CHARACTERISTIC [-> DISPLAY_IDENTIFIER]

  AliasNameAssignment.label (if provided) +
  FlatInstanceDescriptor.(desc |upstreamReference.desc) ->
  MEASUREMENT LongIdentifier
  CHARACTERISTIC LongIdentifier

- If `AliasNameAssignment` references a `SwSystemconstant`:

  AliasNameAssignment.shortLabel ->
  SYSTEM_CONSTANT -> Name for SwSystemconstants

- If `AliasNameAssignment` references a `Unit`:

  AliasNameAssignment.shortLabel ->
  UNIT -> Name for Units

9.6 Variant Handling in ECU Extract

The System Template supports the creation of variants in many of its model elements. Depending on the binding time, some of this variability may have been already resolved within the System Configuration Description at the time of creating the ECU Extract, and a cleanup step may have removed some of the complexity by removing the out-configured variability.

If however binding of a concrete variation condition happens in a later stage of the AUTOSAR methodology (e.g. during ECU Configuration or even post build), or if for other process reasons such a cleanup step is not applicable, the variability needs to be carried over to the ECU Extract.

9.6.1 System Constants

In the AUTOSAR variant handling concept, SwSystemconst represents a variant selector which needs to have its value assigned latest at binding time of any expression which refers to it. Such a value assignment may be done literally using a fixed value, or by specifying a formula, depending on the values of other variant selectors. The elements to do this are collected in a SwSystemconstValueSet, aggregating individual value assignment expressions in the form of SwSystemConstValue.

In the ECU Extract, all SwSystemconst elements are included that influence its variable content. In detail the following rules for the inclusion of SwSystemconst apply:

- ECU Extract shall contain all SwSystemconst elements that are being referenced directly by variable elements contained in the ECU Extract.
- Additionally, whenever a SwSystemconst's value is assigned indirectly using an SwSystemconstValue's ValueByFormula expression, each SwSystemconstValue refered to in the assignment formula needs to be included, too. As such assignments may be nested in multiple levels, the whole directed acyclic graph of SwSystemConst elements influencing the ECU Extract's variability need to be included.

Additionally to the SwSystemconst elements also all relevant SwSystemConstValue assignments need to be included. As they are aggregated by SwSystemconstValueSet, the whole Value Set is included whenever one of its SwSystemconstValue assignments is relevant for the ECU Extract.

Note: Typically, the assignment of Variants ("Binding´´) will be done in a Variant Configuration work product, separate from the actual ECU Extract. In this case, the relevant information from the Variant Configuration also needs to be extracted and delivered in combination with the ECU Extract. From the model point of view it doesn't matter whether ECU Extract and Variant Configuration are contained in the same file or in separate files.

9.6.2 Nested Whole/Part class variants

In case of flattening the hierarchical VFB view to the ECU flat view representation, the case may appear that one conditional SwComponentPrototype is nested within another SwComponentPrototype depending on another variance condition. As the resulting ECU flat view only has a flat representation of SwComponentPrototypes, such a double condition needs to be resolved to a single condition in the resulting `SwComponentPrototypes`.

In this case, the variation condition formula needs to be altered such that the two (or more) individual conditions are combined in a boolean AND function.

— AUTOSAR CONFIDENTIAL —

A Glossary

Artifact This is a Work Product Definition that provides a description and definition for tangible work product types. Artifacts may be composed of other artifacts ([22]).

At a high level, an artifact is represented as a single conceptual file.

AUTOSAR Tool This is a software tool which supports one or more tasks defined as AUTOSAR tasks in the methodology. Depending on the supported tasks, an AUTOSAR tool can act as an authoring tool, a converter tool, a processor tool or as a combination of those (see separate definitions).

AUTOSAR Authoring Tool An AUTOSAR Tool used to create and modify AUTOSAR XML Descriptions. Example: System Description Editor.

AUTOSAR Converter Tool An AUTOSAR Tool used to create AUTOSAR XML files by converting information from other AUTOSAR XML files. Example: ECU Flattener

AUTOSAR Definition This is the definition of parameters which can have values. One could say that the parameter values are Instances of the definitions. But in the meta model hierarchy of AUTOSAR, definitions are also instances of the meta model and therefore considered as a description. Examples for AUTOSAR definitions are: `EcucParameterDef`, `PostBuildVariantCriterion`, `SwSystemconst`.

AUTOSAR XML Description In AUTOSAR this means "filled Template". In fact an AUTOSAR XML description is the XML representation of an AUTOSAR model.

The AUTOSAR XML description can consist of several files. Each individual file represents an AUTOSAR partial model and shall validate successfully against the AUTOSAR XML schema.

AUTOSAR Meta-Model This is an UML2.0 model that defines the language for describing AUTOSAR systems. The AUTOSAR meta-model is an UML representation of the AUTOSAR templates. UML2.0 class diagrams are used to describe the attributes and their interrelationships. Stereotypes, UML tags and OCL expressions (object constraint language) are used for defining specific semantics and constraints.

AUTOSAR Model This is a representation of an AUTOSAR product. The AUTOSAR model represents aspects suitable to the intended use according to the AUTOSAR methodology.

Strictly speaking, this is an instance of the AUTOSAR meta-model. The information contained in the AUTOSAR model can be anything that is representable according to the AUTOSAR meta-model.

AUTOSAR Partial Model In AUTOSAR, the possible partitioning of models is marked in the meta-model by ≪`atpSplitable`≫. One partial model is represented in an AUTOSAR XML description by one file. The partial model does not need to fulfill all semantic constraints applicable to an AUTOSAR model.

AUTOSAR Processor Tool An AUTOSAR Tool used to create non-AUTOSAR files by processing information from AUTOSAR XML files. Example: RTE Generator

AUTOSAR Template The term "Template" is used in AUTOSAR to describe the format different kinds of descriptions. The term template comes from the idea, that AUTOSAR defines a kind of form which shall be filled out in order to describe a model. The filled form is then called the description.

In fact the AUTOSAR templates are now defined as a meta model.

AUTOSAR XML Schema This is a W3C XML schema that defines the language for exchanging AUTOSAR models. This Schema is derived from the AUTOSAR meta model. The AUTOSAR XML Schema defines the AUTOSAR data exchange format.

Blueprint This is a model from which other models can be derived by copy and refinement. Note that in contrast to meta model resp. types, this process is *not* an instantiation.

Instance Generally this is a particular exemplar of a model or of a type.

Meta-Model This defines the building blocks of a model. In that sense, a Meta-Model represents the language for building models.

Meta-Data This includes pertinent information about data, including information about the authorship, versioning, access-rights, timestamps etc.

Model A Model is an simplified representation of reality. The model represents the aspects suitable for an intended purpose.

Partial Model This is a part of a model which is intended to be persisted in one particular artifact.

Pattern in GST : This is an approach to simplify the definition of the meta model by applying a model transformation. This transformation crates an enhanced model out of an annotated model.

Property A property is a structural feature of an object. As an example a “connector” has the properties “receive port” and “send port”

Properties are made variant by the ≪`atpVariation`≫.

Prototype This is the implementation of a role of a type within the definition of another type. In other words a type may contain Prototypes that in turn are typed by "Types". Each one of these prototypes becomes an instance when this type is instantiated.

Type A type provides features that can appear in various roles of this type.

Value This is a particular value assigned to a “Definition”.

Variability Variability of a system is its quality to describe a set of variants. These variants are characterized by variant specific property settings and / or selections.

As an example, such a system property selection manifests itself in a particular "receive port" for a connection.

This is implemented using the ≪atpVariation≫.

Variant A system variant is a concrete realization of a system, so that all its properties have been set respectively selected. The software system has no variability anymore with respect to the binding time.

This is implemented using EvaluatedVariantSet.

Variation Binding A variant is the result of a variation binding process that resolves the variability of the system by assigning particular values/selections to all the system's properties.

This is implemented by VariationPoint.

Variation Binding Time The variation binding time determines the step in the methodology at which the variability given by a set of variable properties is resolved.

This is implementing by vh.LatestBindingtime at the related properties .

Variation Definition Time The variation definition time determines the step in the methodology at which the variation points are defined.

Variation Point A variation point indicates that a property is subject to variation. Furthermore, it is associated with a condition and a binding time which define the system context for the selection / setting of a concrete variant.

This is implemented by VariationPoint.

B Supported special use-cases

The description means of the communication matrix in the System Template potentially support a variety of use-cases. Some combinations of description means are explicitly ruled-out by semantical constraints. But the remaining space for the possible descriptions is so huge, that certain use-cases are actually not supported by tool-vendors because they did not consider them. This chapter describes special use-cases that can be specified in the System Template in order to get a harmonized support by tools.

B.1 Support of sending / receiving same Can/Flexray Frame on same channel

Description: The System Template supports the definition of a communication where the same Can/Flexray frame is sent and received on the same channel of one ECU.

Rationale: This use-case occurs in gateway ECUs which are used in several vehicle platforms.

Implementation: This usage shall be supported by defining one `Frame` and one `FrameTriggering` with different directions on the referenced `FramePorts` for the same channel. Also one `Pdu` and one `PduTriggering` with different directions on the referenced `IPduPorts` for the same channel shall be used.

Example: In figure B.1 a sample network setup is shown. The ECU1 is designed to send the Frame_X on the channel. The ECU2, ECU3 and ECU4 do receive the information. But since ECU1 is optional, ECU4 is also designed to send the Frame_X on the network (in case ECU1 is not present).

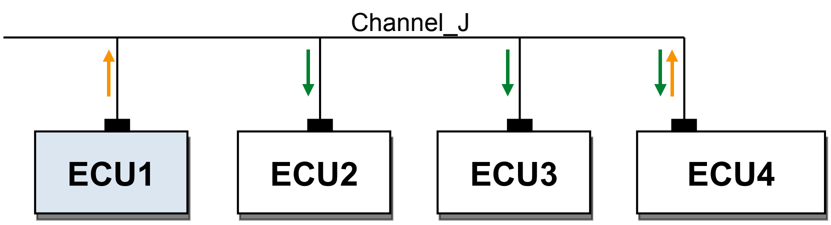

Figure B.1: Example of network setup with one Frame being received and sent on the same ECU and channel

In the system description there exists one definition for the Frame_X and one FrameTriggering for the channel (figure B.2). Each ECU sending or receiving the frame does define one FramePort per direction, thus for ECU4 there are two FramePorts defined.

For each Pdu mapped to the frame there exists one definition for the Pdu_X and one PduTriggering for the channel. Each ECU sending or receiving the Pdu does define one IPduPort per direction, thus for ECU4 there are two IPduPorts defined.

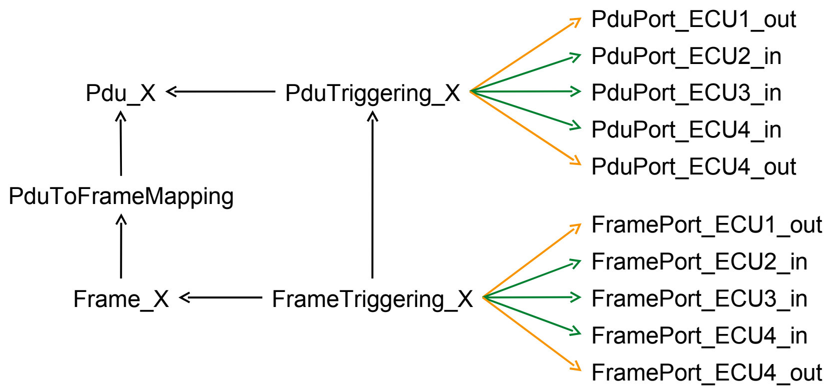

Figure B.2: Structure to reflect the frame- and pdu-triggering setup of one Frame being received and sent on the same ECU and channel

In case a System Extract / ECU Extract is build, only the relevant FramePorts and IPduPorts for the corresponding ECU are extracted. Especially in case an additional ECU is designed to send and receive the same Frame all the other ECU extracts will not be affected by this change.

B.2 Support of Frames, Pdus and Signals with length 0

The AUTOSAR client-server communication requires to support signals with length zero. If no actual data is configured for a client-server communication, i. e. the applicable `ClientServerToSignalGroupMapping` owns only an emptySignal, the RTE sends a signal group with an emptySignal to initiate the communication. In this case the element `EmptySignalMapping` in the `ClientServerToSignalGroupMapping` shall reference a `SystemSignal` that is referenced by an `ISignal` with length equal to zero. Such empty `ISignals` will be mapped into `Pdus` and `Frames` and therefore `Pdus` and `Frames` with length zero are also supported by the System Template.

C Detailed Representation of InstanceRef Associations in the System Template

As a special type of association "instanceRef" refers to an exact instance of the referenced class, requiring additional information of the target and the context. This is explained in detail in the AUTOSAR Generic Structure Template [1]. This chapter contains the detailed InstanceRef Diagrams.

C.1 Usage of InstanceRefs in Data Mapping diagrams

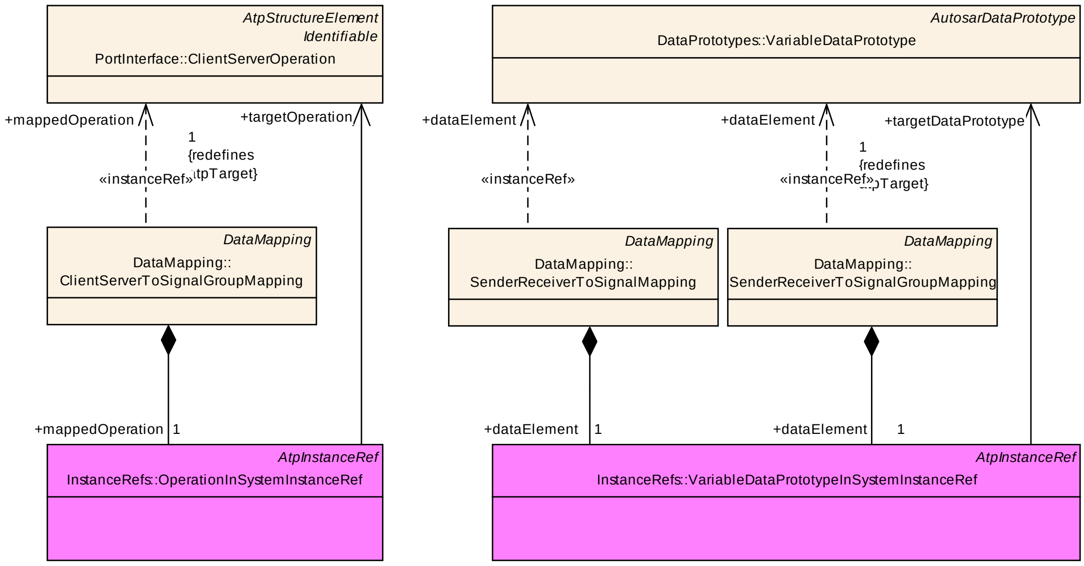

Figure C.1: Data Mapping Instance Ref Usage

C.2 Usage of InstanceRefs in SW Mapping diagrams

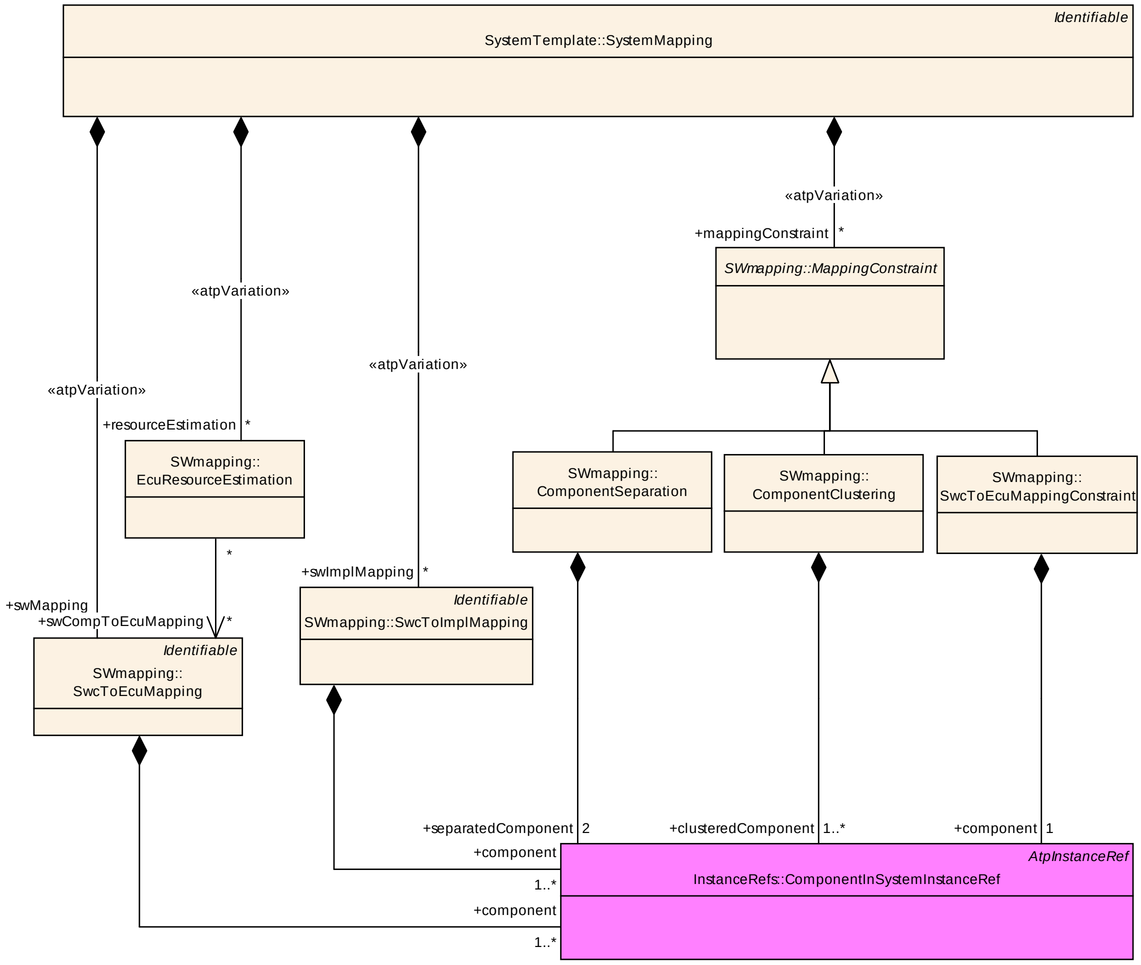

Figure C.2: SW Mapping Instance Ref Usage

C.3 Usage of InstanceRefs in Signal Path Constraint diagrams

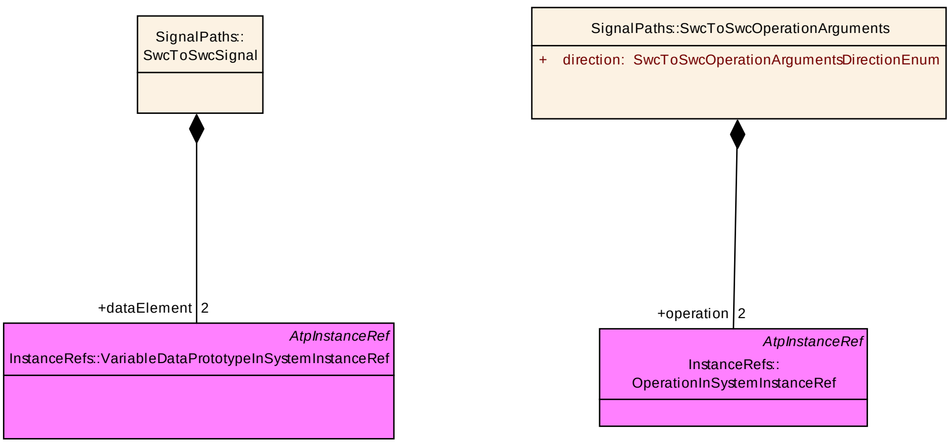

Figure C.3: SW Mapping Instance Ref Usage

C.4 Usage of InstanceRefs in PncMapping

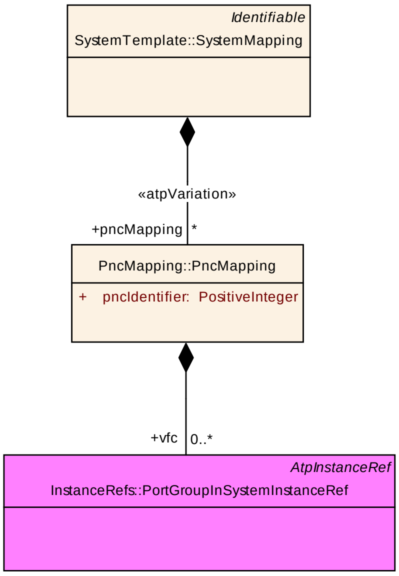

Figure C.4: Partial Network Mapping Instance Ref Usage

C.5 "SWC in System" InstanceRef

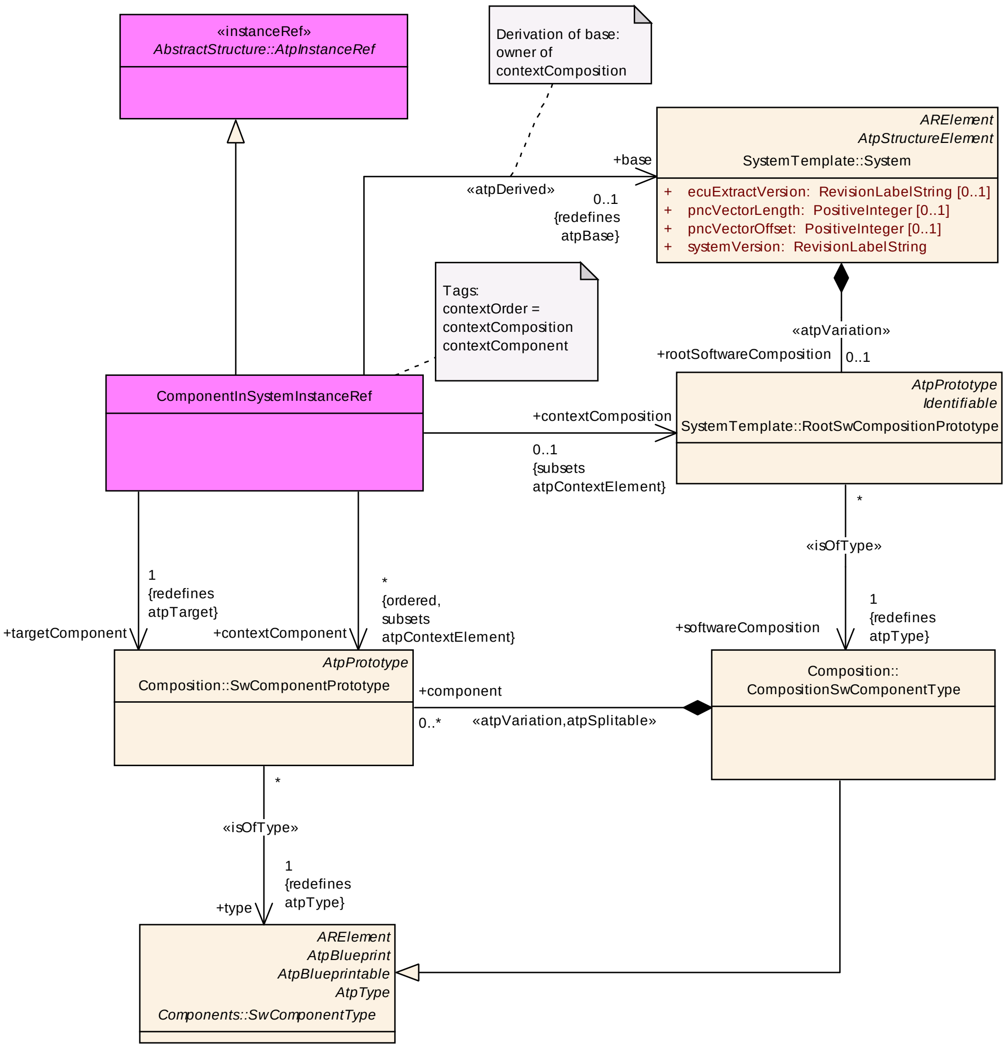

Figure C.5: ComponentInSystem InstanceRef

If the referenced `SwComponentPrototype` is located within the `RootSwComposition` of a `System` then the base context reference and the context reference to the `RootSwComposition` shall be provided. If the referenced `SwComponentPrototype` is the `RootSwComposition` itself then the base context reference and the context reference to the `RootSwComposition` shall be skipped and only the target reference to the `SwComponentPrototype` shall be used.

C.6 "Operation in System" InstanceRef

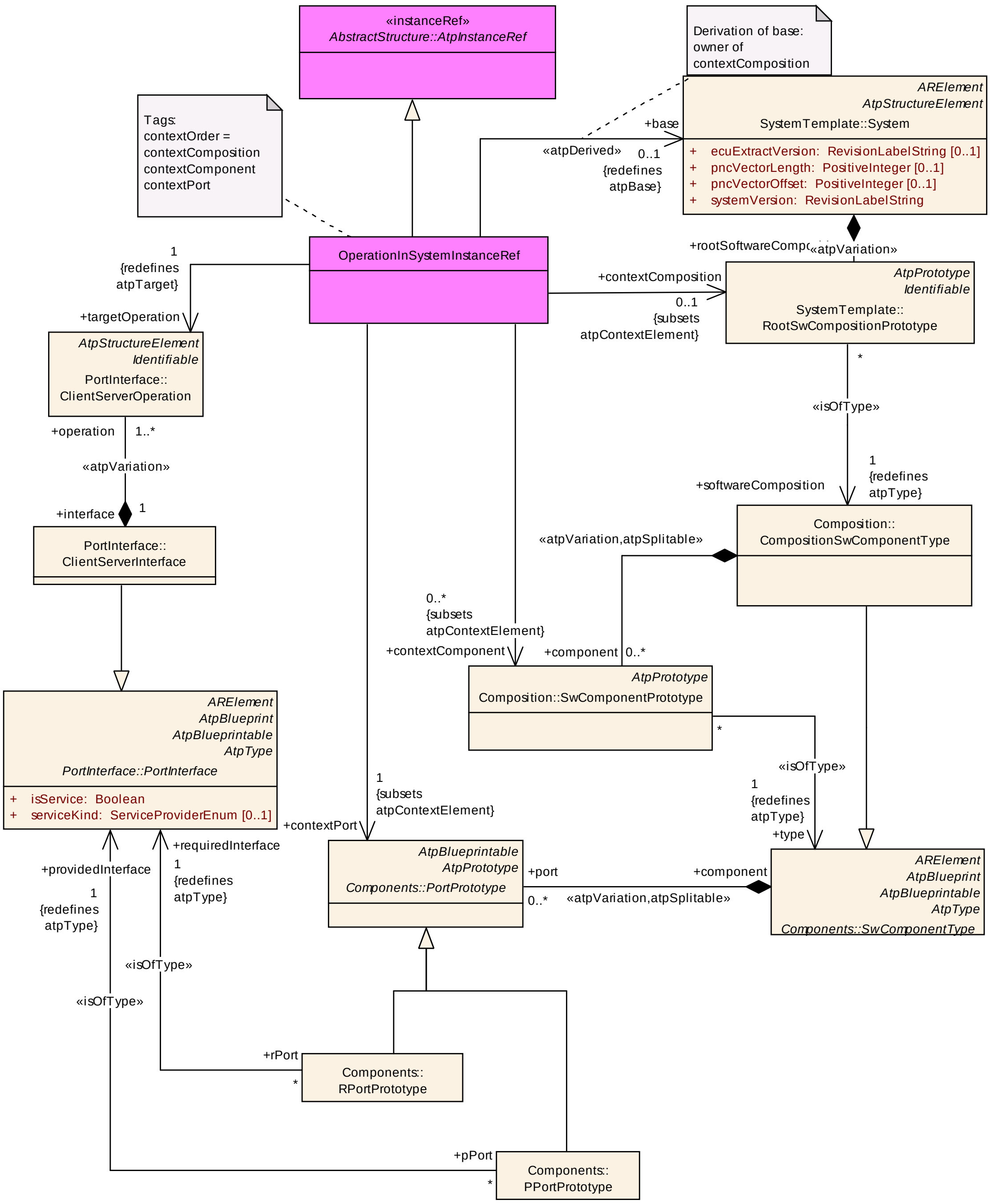

Figure C.6: OperationInSystem InstanceRef

If the referenced `ClientServerOperation` is part of a `PortInterface` of a software component that is located within the `RootSwComposition` then the base context reference and the `contextComposition` reference to the `RootSwComposition` shall be provided. If the referenced `ClientServerOperation` is part of a `PortInterface` of the `RootSwComposition` itself then the base context reference and the `contextComposition` reference to the `RootSwComposition` shall be skipped and the `RootSwComposition` shall be referenced as `ContextComponent`.

C.7 "VariableDataPrototype" InstanceRef

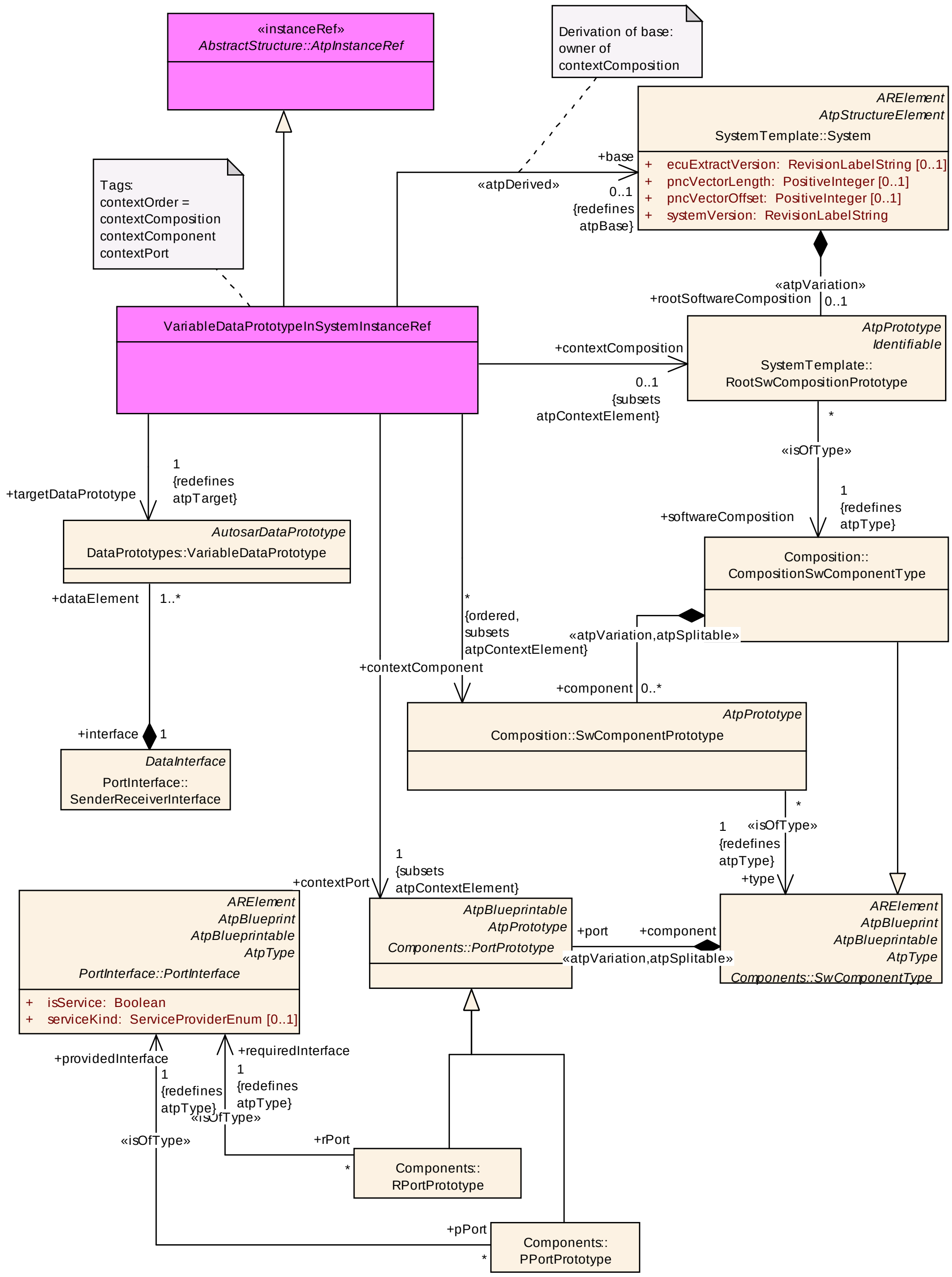

Figure C.7: VariableDataPrototypeInSystem InstanceRef

If the referenced `VariableDataPrototype` is part of a `PortInterface` of a software component that is located within the `RootSwComposition` then the base context reference and the `contextComposition` reference to the `RootSwComposition` shall be provided. If the referenced `VariableDataPrototype` is part of a `PortInterface` of the `RootSwComposition` itself then the base context reference and the `contextComposition` reference to the `RootSwComposition` shall be skipped and the `RootSwComposition` shall be referenced as `ContextComponent`.

C.8 "PortGroup in System" InstanceRef

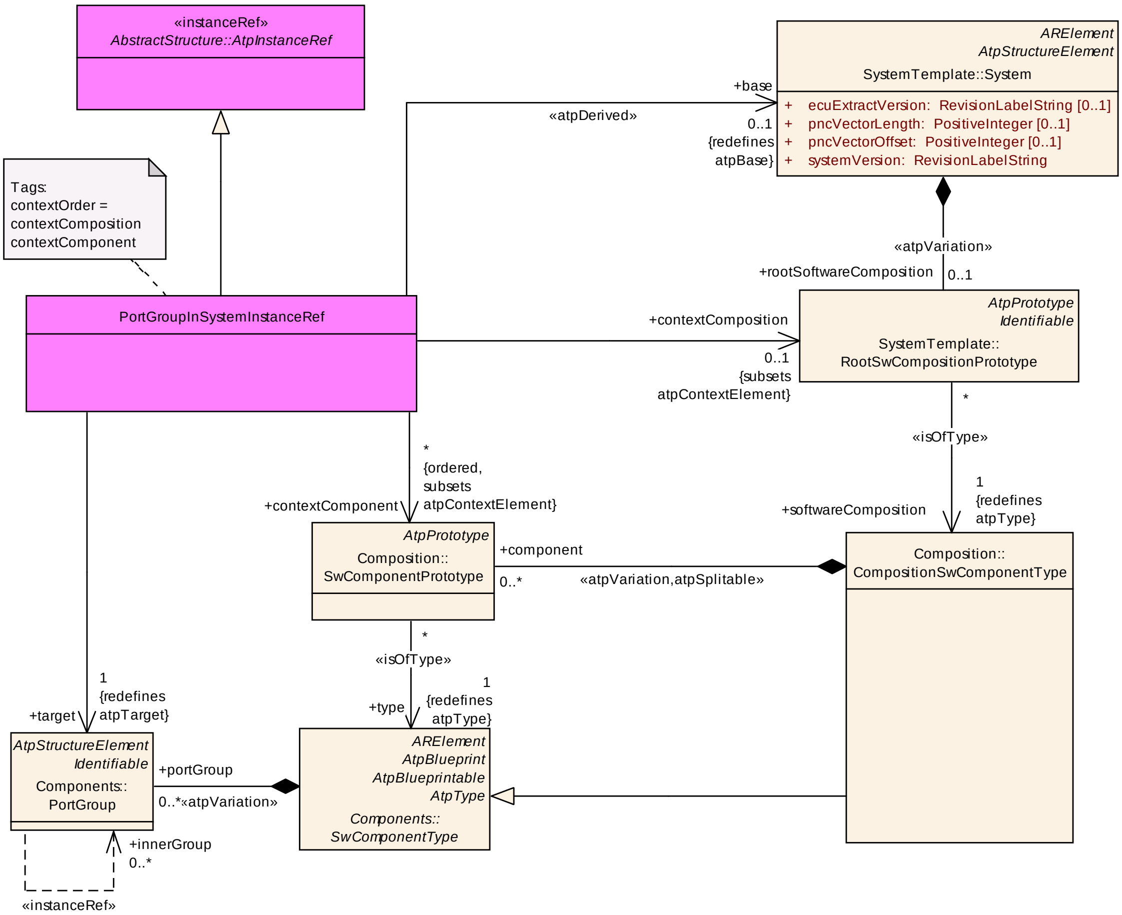

Figure C.8: PortGroupInSystem InstanceRef

If the referenced `PortGroup` is part of a software component that is located within the `RootSwComposition` then the base context reference and the `contextComposition` reference to the `RootSwComposition` shall be provided. If the referenced `PortGroup` is part of the `RootSwComposition` itself then the base context reference and the `contextComposition` reference to the `RootSwComposition` shall be skipped and the `RootSwComposition` shall be referenced as `ContextComponent`.

D Harmonisation between Upstream Templates and ECU Configuration

This chapter describes the mapping of the ECU Configuration parameters (M1 model) onto the classes and attributes of the AUTOSAR upstream templates (System Template, SW Component Template and ECU Resource Template). The relationships between upstream templates and ECU Configuration are described in order to answer typical questions like: How shall a supplier use the information in a System Description in order to fulfill the needs defined by the systems engineer? How is a tool vendor suppose to generate an ECU Configuration Description out of ECU Extract of System Description?

The tables contain the following columns:

bsw module: Name of BSW module

bsw context: Reference to parameter container

bsw type: Type of parameter

bsw param: Name of the BSW parameter

bsw desc: Description from the configuration document

m2 template: System Template, SW Component Template, ECU Resource Template

m2 param: Name of the upstream template parameter

m2 desc: Description from the upstream template definition

mapping rule: Textual description on how to transform between M2 and BSW domains

mapping type:

- local: no mapping needed since parameter local to BSW
- partial: some data can be automatically mapped but not all
- full: all data can be automatically mapped

D.1 Can Driver Mapping

BSW Module	BSW Context	
Can	Can	
BSW Parameter		**BSW Type**
CanConfigSet		EcucParamConfContainerDef
BSW Description		
This is the multiple configuration set container for CAN Driver		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
Can	Can/CanConfigSet	
BSW Parameter		**BSW Type**
CanController		EcucParamConfContainerDef
BSW Description		
This container contains the configuration parameters of the CAN controller(s).		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
Can	Can/CanConfigSet/CanController	
BSW Parameter		**BSW Type**
CanBusoffProcessing		EcucEnumerationParamDef
BSW Description		
Enables / disables API Can_MainFunction_BusOff() for handling busoff events in polling mode.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
Can	Can/CanConfigSet/CanController	
BSW Parameter		**BSW Type**
CanControllerActivation		EcucBooleanParamDef
BSW Description		
Defines if a CAN controller is used in the configuration.		
M2 Template	**M2 Description**	
M2 Parameter		

Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**	
Can	Can/CanConfigSet/CanController	
BSW Parameter		**BSW Type**
CanControllerBaseAddress		EcucIntegerParamDef
BSW Description		
Specifies the CAN controller base address.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
Can	Can/CanConfigSet/CanController	
BSW Parameter		**BSW Type**
CanControllerBaudrateConfig		EcucParamConfContainerDef
BSW Description		
This container contains bit timing related configuration parameters of the CAN controller(s).		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
Can	Can/CanConfigSet/CanController/CanControllerBaudrateConfig	
BSW Parameter		**BSW Type**
CanControllerBaudRate		EcucIntegerParamDef
BSW Description		
Specifies the baudrate of the controller in kbps.		
M2 Template	**M2 Description**	
SystemTemplate	channels speed in bits per second	
M2 Parameter		
Fibex::FibexCore::CoreTopology::CommunicationCluster.speed		
Mapping Rule		**Mapping Type**
SystemTemplate speed is in bps, so divide it by 1000 to get kbps		full

BSW Module	**BSW Context**	
Can	Can/CanConfigSet/CanController/CanControllerBaudrateConfig	
BSW Parameter		**BSW Type**
CanControllerPropSeg		EcucIntegerParamDef
BSW Description		
Specifies propagation delay in time quantas.		

M2 Template	M2 Description	
SystemTemplate		
M2 Parameter		
Fibex::Fibex4Can::CanTopology::CanControllerConfiguration.timeSeg1 - Fibex::Fibex4Can::CanTopology::CanControllerConfiguration.timeSeg2		
Mapping Rule		**Mapping Type**
PropSeg = timeSeg1 - timeSeg2		full

BSW Module	BSW Context	
Can	Can/CanConfigSet/CanController/CanControllerBaudrateConfig	
BSW Parameter		**BSW Type**
CanControllerSeg1		EcucIntegerParamDef
BSW Description		
Specifies phase segment 1 in time quantas.		
M2 Template	**M2 Description**	
SystemTemplate	The number of quanta after the sampling point	
M2 Parameter		
Fibex::Fibex4Can::CanTopology::CanControllerConfiguration.timeSeg1		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
Can	Can/CanConfigSet/CanController/CanControllerBaudrateConfig	
BSW Parameter		**BSW Type**
CanControllerSeg2		EcucIntegerParamDef
BSW Description		
Specifies phase segment 2 in time quantas.		
M2 Template	**M2 Description**	
SystemTemplate	The number of quanta after the sampling point	
M2 Parameter		
Fibex::Fibex4Can::CanTopology::CanControllerConfiguration.timeSeg2		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
Can	Can/CanConfigSet/CanController/CanControllerBaudrateConfig	
BSW Parameter		**BSW Type**
CanControllerSyncJumpWidth		EcucIntegerParamDef
BSW Description		
Specifies the synchronization jump width for the controller in time quantas.		
M2 Template	**M2 Description**	
System Template	The number of quanta in the Synchronization Jump Width, SJW.	
M2 Parameter		
Fibex::Fibex4Can::CanTopology::CanCommunicationCluster.syncJumpWidth		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context
Can	Can/CanConfigSet/CanController

— AUTOSAR CONFIDENTIAL —

BSW Parameter		BSW Type	
CanControllerDefaultBaudrate		EcucReferenceDef	
BSW Description			
Reference to baudrate configuration container configured for the Can Controller.			
M2 Template	**M2 Description**		
SystemTemplate	channels speed in bits per second		
M2 Parameter			
Fibex::FibexCore::CoreTopology::CommunicationCluster.speed			
Mapping Rule			**Mapping Type**
SystemTemplate speed is in bps, so divide it by 1000 to get kbps			full

BSW Module	BSW Context		
Can	Can/CanConfigSet/CanController		
BSW Parameter		**BSW Type**	
CanControllerId		EcucIntegerParamDef	
BSW Description			
This parameter provides the controller ID which is unique in a given CAN Driver. The value for this parameter starts with 0 and continue without any gaps.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	BSW Context		
Can	Can/CanConfigSet/CanController		
BSW Parameter		**BSW Type**	
CanCpuClockRef		EcucReferenceDef	
BSW Description			
Reference to the CPU clock configuration, which is set in the MCU driver configuration			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	BSW Context		
Can	Can/CanConfigSet/CanController		
BSW Parameter		**BSW Type**	
CanFilterMask		EcucParamConfContainerDef	
BSW Description			
This container contains the configuration (parameters) of the CAN Filter Mask(s).			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

— AUTOSAR CONFIDENTIAL —

BSW Module	BSW Context	
Can	Can/CanConfigSet/CanController/CanFilterMask	
BSW Parameter		**BSW Type**
CanFilterMaskValue		EcucIntegerParamDef
BSW Description		
Describes a mask for hardware-based filtering of CAN identifiers. The CAN identifiers of incoming messages are masked with the appropriate CanFilterMaskValue. Bits holding a 0 mean don't care, i.e. do not compare the message's identifier in the respective bit position. The mask shall be build by filling with leading 0. In case of CanIdType EXTENDED or MIXED a 29 bit mask shall be build. In case of CanIdType STANDARD a 11 bit mask shall be build		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
Can	Can/CanConfigSet/CanController	
BSW Parameter		**BSW Type**
CanRxProcessing		EcucEnumerationParamDef
BSW Description		
Enables / disables API Can_MainFunction_Read() for handling PDU reception events in polling mode.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
Can	Can/CanConfigSet/CanController	
BSW Parameter		**BSW Type**
CanTTController		EcucParamConfContainerDef
BSW Description		
This container is only included and valid if TTCAN SWS is used and TTCAN is enabled. This container contains the configuration parameters of the TTCAN controller(s) (which are needed in addition to the configuration parameters of the CAN controller(s)). CanTTController is only included, if the controller supports TTCAN.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**

BSW Module	BSW Context	
Can	Can/CanConfigSet/CanController/CanTTController	
BSW Parameter		**BSW Type**
CanTTControllerApplWatchdogLimit		EcucIntegerParamDef
BSW Description		
Defines the maximum time period (unit is 256 times NTU) after which the application has to serve the watchdog.		
M2 Template	**M2 Description**	
SystemTemplate	The Appl_Watchdog_Limit shall be an 8-bit value specifying the period for the application watchdog in Appl_Watchdog_Limit times 256 NTUs.	
M2 Parameter		
TtcanTopology::TtcanCommunicationController.applWatchdogLimit		
Mapping Rule		**Mapping Type**
1:1 mapping		local

BSW Module	BSW Context	
Can	Can/CanConfigSet/CanController/CanTTController	
BSW Parameter		**BSW Type**
CanTTControllerCycleCountMax		EcucIntegerParamDef
BSW Description		
Defines the value for cycle_count_max. Allowed values: 0x00: 1 basic cycle 0x01: 2 basic cycles 0x03: 4 basic cycles 0x07: 8 basic cycles 0x0F: 16 basic cycles 0x1F: 32 basic cycles 0x3F: 64 basic cycles		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
Can	Can/CanConfigSet/CanController/CanTTController	
BSW Parameter		**BSW Type**
CanTTControllerExpectedTxTrigger		EcucIntegerParamDef
BSW Description		
Number of expected_tx_trigger.		
M2 Template	**M2 Description**	
SystemTemplate	The Expected_Tx_Trigger shall be an eight (8) bit value which limits the number of messages the FSE may try to transmit in one matrix cycle.	
M2 Parameter		
TtcanTopology::TtcanCommunicationController.expectedTxTrigger		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context

Can	Can/CanConfigSet/CanController/CanTTController		
BSW Parameter		**BSW Type**	
CanTTControllerExternalClockSynchronisation		EcucBooleanParamDef	
BSW Description			
Enables/disables the external clock synchronization. TRUE: External clock synchronization enabled. FALSE: External clock synchronization disabled. This parameter shall only be configurable if parameter CanTTControllerLevel2 equals TRUE.			
M2 Template	**M2 Description**		
SystemTemplate	One bit shall be used to configure whether or not external clock synchronisation will be allowed during runtime (only Level 2).		
M2 Parameter			
TtcanTopology::TtcanCommunicationController.externalClockSynchronisation			
Mapping Rule			**Mapping Type**
1:1 mapping			full

BSW Module	**BSW Context**		
Can	Can/CanConfigSet/CanController/CanTTController		
BSW Parameter		**BSW Type**	
CanTTControllerGlobalTimeFiltering		EcucBooleanParamDef	
BSW Description			
Enables/disables the global time filtering. TRUE: Global time filtering enabled. FALSE: Global time filtering disabled. This parameter shall only be configurable if parameter CanTTControllerLevel2 equals TRUE.			
M2 Template	**M2 Description**		
	Enables/disables global time filtering		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	**BSW Context**		
Can	Can/CanConfigSet/CanController/CanTTController		
BSW Parameter		**BSW Type**	
CanTTControllerInitialRefOffset		EcucIntegerParamDef	
BSW Description			
Defines the initial value for ref trigger offset.			
M2 Template	**M2 Description**		
SystemTemplate	The Initial_Ref_Offset shall be an eight (8) bit value for the initialisation of Ref_Trigger_Offset.		
M2 Parameter			
TtcanTopology::TtcanCommunicationController.initialRefOffset			
Mapping Rule			**Mapping Type**
1:1 mapping			full

BSW Module	BSW Context	
Can	Can/CanConfigSet/CanController/CanTTController	
BSW Parameter		**BSW Type**
CanTTControllerInterruptEnable		EcucIntegerParamDef
BSW Description		
Enables/disables the respective interrupts. Bit Position set to 1: Enable respective interrupt. Bit Position set to 0: Disable respective interrupt. Bit Position / Interrupt Source: 10: Application Watchdog. 9: Watch Trigger reached. 8: Initialization Watch Trigger reached. 7: Change of Error Level. 6: Tx Overflow. 5: Tx Underflow. 4: Global Time Error. 3: Gap. 2: Start of Cycle. 1: Time Discontinuity. 0: Master State Change. Bit position "1: Time Discontinuity" and "4: Global Time Error" shall only be configurable if parameter CanTTControllerLevel2 equals TRUE.		
M2 Template	**M2 Description**	
	Enables/disables interrupts	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
Can	Can/CanConfigSet/CanController/CanTTController	
BSW Parameter		**BSW Type**
CanTTControllerLevel2		EcucBooleanParamDef
BSW Description		
Defines whether Level 2 or Level 1 is used. TRUE: Level 2. FALSE: Level 1. If this parameter is set to FALSE then all parameters with dependency to CanTTControllerLevel2 need not be configured.		
M2 Template	**M2 Description**	
SystemTemplate	One bit shall be used to distinguish between Level 1 and Level 2.	
M2 Parameter		
TtcanTopology::TtcanCommunicationController.timeTriggeredCanLevel		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
Can	Can/CanConfigSet/CanController/CanTTController	
BSW Parameter		**BSW Type**

CanTTControllerNTUConfig		EcucFloatParamDef	
BSW Description			
Defines the config value for NTU (network time unit). Value given in microseconds. The value configured shall be greater than 0. Together with the local oscillator period, the TUR (time unit ratio) can be derived from the NTU. This parameter shall only be configurable if parameter CanTTControllerLevel2 equals TRUE.			
M2 Template	**M2 Description**		
SystemTemplate	Unit measuring all times and providing a constant of the whole network. For level 1, this is always the CAN bit time. Unit: seconds.		
M2 Parameter			
TtcanTopology::TtcanCluster.ntu			
Mapping Rule			**Mapping Type**
NTU = system clock period x (TUR Numerator / TUR Denominator)			full

BSW Module	**BSW Context**		
Can	Can/CanConfigSet/CanController/CanTTController		
BSW Parameter		**BSW Type**	
CanTTControllerOperationMode		EcucEnumerationParamDef	
BSW Description			
Defines the operation mode.			
M2 Template	**M2 Description**		
SystemTemplate	Operation mode: Time-triggered or event synchronized time-triggered		
M2 Parameter			
TtcanTopology::TtcanCluster.operationMode			
Mapping Rule			**Mapping Type**
1:1 mapping			full

BSW Module	**BSW Context**		
Can	Can/CanConfigSet/CanController/CanTTController		
BSW Parameter		**BSW Type**	
CanTTControllerSyncDeviation		EcucFloatParamDef	
BSW Description			
Defines the maximum synchronization deviation: Given as a percentage value of the NTU (network time unit). The value configured shall be greater than 0. This parameter shall only be configurable if parameter CanTTControllerLevel2 equals TRUE.			
M2 Template	**M2 Description**		
	Defines maximum tolerated synchronization deviation		
M2 Parameter			
Mapping Rule			**Mapping Type**
Synchronisation Deviation <= 2ˆ (CanTTSyncDeviation + 5).			local

BSW Module	**BSW Context**		
Can	Can/CanConfigSet/CanController/CanTTController		
BSW Parameter		**BSW Type**	
CanTTControllerTURRestore		EcucBooleanParamDef	
BSW Description			

Enables/disables the TUR restore. Note that the value configured for TUR can be derived from the value configured for NTU and the local oscillator preriod. TRUE: TUR restore enabled. FALSE: TUR restore disabled. This parameter shall only be configurable if parameter CanTTControllerLevel2 equals TRUE.		
M2 Template	**M2 Description**	
	Enables/disables TUR restore	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
Can	Can/CanConfigSet/CanController/CanTTController	
BSW Parameter		**BSW Type**
CanTTControllerTimeMaster		EcucBooleanParamDef
BSW Description		
Defines whether the controller acts as a potential time master. TRUE: Potential time master. FALSE: Time slave.		
M2 Template	**M2 Description**	
SystemTemplate	Master-salve mode: Potential time master or slave	
M2 Parameter		
TtcanTopology::TtcanCommunicationController.master		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
Can	Can/CanConfigSet/CanController/CanTTController	
BSW Parameter		**BSW Type**
CanTTControllerTimeMasterPriority		EcucIntegerParamDef
BSW Description		
Defines the time master priority.		
M2 Template	**M2 Description**	
SystemTemplate	The time master priority shall contain a three bit value for the priority of the current time master (the last three bits of the identifier of the reference message). This can be derived from the frame-triggering's triggers.	
M2 Parameter		
TtcanTopology::TtcanCommunicationController.timeMasterPriority		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
Can	Can/CanConfigSet/CanController/CanTTController	
BSW Parameter		**BSW Type**
CanTTControllerTxEnableWindowLength		EcucIntegerParamDef
BSW Description		

Length of the tx enable window given in CAN bit times. Definition parameter "CanTTControllerTxEnableWindowlength" is used such that: Length of enable window = CanTTControllerTxEnableWindowLength + 1		
M2 Template	**M2 Description**	
SystemTemplate	Length of the tx enable window given in CAN bit times	
M2 Parameter		
TtcanTopology::TtcanCommunicationController.txEnableWindowLength		
Mapping Rule		**Mapping Type**
Length of enable window = CanTTControllerTxEnableWindowLength + 1		full

BSW Module	**BSW Context**		
Can	Can/CanConfigSet/CanController/CanTTController		
BSW Parameter		**BSW Type**	
CanTTControllerWatchTriggerGapTimeMark		EcucIntegerParamDef	
BSW Description			
watch trigger time mark after a gap			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	**BSW Context**		
Can	Can/CanConfigSet/CanController/CanTTController		
BSW Parameter		**BSW Type**	
CanTTControllerWatchTriggerTimeMark		EcucIntegerParamDef	
BSW Description			
watch trigger time mark			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	**BSW Context**		
Can	Can/CanConfigSet/CanController/CanTTController		
BSW Parameter		**BSW Type**	
CanTTIRQProcessing		EcucEnumerationParamDef	
BSW Description			
Enables / disables API Can_MainFunction_BusOff() for handling busoff events in polling mode.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	BSW Context	
Can	Can/CanConfigSet/CanController	
BSW Parameter		**BSW Type**
CanTxProcessing		EcucEnumerationParamDef
BSW Description		
Enables / disables API Can_MainFunction_Write() for handling PDU transmission events in polling mode.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
Can	Can/CanConfigSet/CanController	
BSW Parameter		**BSW Type**
CanWakeupProcessing		EcucEnumerationParamDef
BSW Description		
Enables / disables API Can_MainFunction_Wakeup() for handling wakeup events in polling mode.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
Can	Can/CanConfigSet/CanController	
BSW Parameter		**BSW Type**
CanWakeupSourceRef		EcucSymbolicNameReferenceDef
BSW Description		
This parameter contains a reference to the Wakeup Source for this controller as defined in the ECU State Manager. Implementation Type: reference to EcuM_WakeupSourceType		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
Can	Can/CanConfigSet/CanController	
BSW Parameter		**BSW Type**
CanWakeupSupport		EcucBooleanParamDef
BSW Description		
CAN driver support for wakeup over CAN Bus.		
M2 Template	**M2 Description**	

M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**	
Can	Can/CanConfigSet	
BSW Parameter		**BSW Type**
CanHardwareObject		EcucParamConfContainerDef
BSW Description		
This container contains the configuration (parameters) of CAN Hardware Objects.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
Can	Can/CanConfigSet/CanHardwareObject	
BSW Parameter		**BSW Type**
CanControllerRef		EcucReferenceDef
BSW Description		
Reference to CAN Controller to which the HOH is associated to.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
Can	Can/CanConfigSet/CanHardwareObject	
BSW Parameter		**BSW Type**
CanFilterMaskRef		EcucReferenceDef
BSW Description		
Reference to the filter mask that is used for hardware filtering together with the CAN_ID_VALUE. Different CanHardwareObjects with different CanIdTypes (STANDARD, MIXED, EXTENDED) can share the same CanFilterMask (i.e., the CanFilterMaskRef parameters of these CanHardwareObjects reference the very same CanFilterMask container). This shall be allowed and must be supported by the configuration generators. The CanFilterMaskRef is omitted for 1) CanHardwareObjects with CanObjectType set to TRANSMIT 2) CanHardwareObjects with CanObjectType set to RECEIVE if only a single Can ID shall be received via this CanHardwareObjects (i.e., exact match with CanIdValue)		
M2 Template	**M2 Description**	

M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	BSW Context	
Can	Can/CanConfigSet/CanHardwareObject	
BSW Parameter		**BSW Type**
CanHandleType		EcucEnumerationParamDef
BSW Description		
Specifies the type (Full-CAN or Basic-CAN) of a hardware object.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
Can	Can/CanConfigSet/CanHardwareObject	
BSW Parameter		**BSW Type**
CanIdType		EcucEnumerationParamDef
BSW Description		
Specifies whether the IdValue is of type - standard identifier - extended identifier - mixed mode ImplementationType: Can_IdType		
M2 Template	**M2 Description**	
SystemTemplate	... two types of frame formats. The standard frame format uses 11-bit identifiers ... the extended frame format allows 29-bit identifiers ...	
M2 Parameter		
Fibex::Fibex4Can::CanCommunication::CanFrameTriggering.canAddressingMode		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
Can	Can/CanConfigSet/CanHardwareObject	
BSW Parameter		**BSW Type**
CanIdValue		EcucIntegerParamDef
BSW Description		
Specifies (together with the filter mask) the identifiers range that passes the hardware filter.		
M2 Template	**M2 Description**	
SystemTemplate	To describe a frames identifier on the communication system, usualy with a fixed identifierValue.	
M2 Parameter		
Fibex::Fibex4Can::CanCommunication::CanFrameTriggering.identifier		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
Can	Can/CanConfigSet/CanHardwareObject	
BSW Parameter		**BSW Type**
CanMainFunctionRWPeriodRef		EcucReferenceDef
BSW Description		
Reference to CAN Controller to which the HOH is associated to.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
Can	Can/CanConfigSet/CanHardwareObject	
BSW Parameter		**BSW Type**
CanObjectId		EcucIntegerParamDef
BSW Description		
Holds the handle ID of HRH or HTH. The value of this parameter is unique in a given CAN Driver, and it should start with 0 and continue without any gaps. The HRH and HTH Ids are defined under two different name-spaces. Example: HRH0-0, HRH1-1, HTH0-2, HTH1-3		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
Can	Can/CanConfigSet/CanHardwareObject	
BSW Parameter		**BSW Type**
CanObjectType		EcucEnumerationParamDef
BSW Description		
Specifies if the HardwareObject is used as Transmit or as Receive object		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
Can	Can/CanConfigSet/CanHardwareObject	
BSW Parameter		**BSW Type**
CanTTHardwareObjectTrigger		EcucParamConfContainerDef
BSW Description		

This container is only included and valid if TTCAN SWS is used and TTCAN is enabled. This container contains the configuration (parameters) of TTCAN triggers for Hardware Objects, which are additional to the configuration (parameters) of CAN Hardware Objects. CanTTHardwareObjectTrigger is only included, if the controller supports TTCAN.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**

BSW Module	**BSW Context**	
Can	Can/CanConfigSet/CanHardwareObject/CanTTHardwareObjectTrigger	
BSW Parameter		**BSW Type**
CanTTHardwareObjectBaseCycle		EcucIntegerParamDef
BSW Description		
Defines the cycle_offset. CanTTHardwareObjectBaseCycle must be not greater than cycle_count_max.		
M2 Template	**M2 Description**	
SystemTemplate	The first communication cycle where the frame is sent.This value is incremented at the beginning of each new cycle, ranging from 0 to 63, and is reset to 0 after a sequence of 64 cycles.	
M2 Parameter		
CoreTopology::CycleRepetition.BaseCycle		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
Can	Can/CanConfigSet/CanHardwareObject/CanTTHardwareObjectTrigger	
BSW Parameter		**BSW Type**
CanTTHardwareObjectCycleRepetition		EcucIntegerParamDef
BSW Description		
Defines the repeat_factor. CanTTHardwareObjectCycleRepetition shall be a power of two (2), greater than cycle_offset but not greater than cycle_count_max + 1.		
M2 Template	**M2 Description**	
SystemTemplate	The number of communication cycles (after the first cycle) whenever the frame described by this timing is sent again.	
M2 Parameter		
CoreTopology::CycleRepetition.CycleRepetition		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
Can	Can/CanConfigSet/CanHardwareObject/CanTTHardwareObjectTrigger	
BSW Parameter		**BSW Type**
CanTTHardwareObjectTimeMark		EcucIntegerParamDef
BSW Description		

Defines the point in time, when the trigger will be activated. Value is given in cycle time.	
M2 Template	**M2 Description**
	Time mark of trigger
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**	
Can	Can/CanConfigSet/CanHardwareObject/CanTTHardwareObjectTrigger	
BSW Parameter		**BSW Type**
CanTTHardwareObjectTriggerId		EcucIntegerParamDef
BSW Description		
Sequential number which allows separation of different TTCAN triggers configured for one and the same hardware object.		
M2 Template	**M2 Description**	
	parameter for separation of different triggers defined for one and the same hardware object	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
Can	Can/CanConfigSet/CanHardwareObject/CanTTHardwareObjectTrigger	
BSW Parameter		**BSW Type**
CanTTHardwareObjectTriggerType		EcucEnumerationParamDef
BSW Description		
Defines the type of the trigger associated with the hardware object. This parameter depends on plain CAN parameter CAN_OBJECT_TYPE. If CAN_OBJECT_TYPE equals RECEIVE than this parameter is fixed to CAN_TT_RX_TRIGGER. If CAN_OBJECT_TYPE equals TRANSMIT than one of the following literals is configurable: CAN_TT_TX_REF_TRIGGER, CAN_TT_TX_REF_TRIGGER_GAP, CAN_TT_TX_TRIGGER_MERGED, CAN_TT_TX_TRIGGER_SINGLE, CAN_TT_TX_TRIGGER_EXCLUSIVE.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
Can	Can	
BSW Parameter		**BSW Type**
CanGeneral		EcucParamConfContainerDef
BSW Description		
This container contains the parameters related each CAN Driver Unit.		

M2 Template	M2 Description	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
Can	Can/CanGeneral	
BSW Parameter		**BSW Type**
CanChangeBaudrateApi		EcucBooleanParamDef
BSW Description		
The support of the Can_ChangeBaudrate API is optional. If this parameter is set to true the Can_ChangeBaudrate API shall be supported. Otherwise the API is not supported.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
Can	Can/CanGeneral	
BSW Parameter		**BSW Type**
CanCounterRef		EcucReferenceDef
BSW Description		
This parameter contains a reference to the counter, which is used by the CAN driver.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
Can	Can/CanGeneral	
BSW Parameter		**BSW Type**
CanDevErrorDetection		EcucBooleanParamDef
BSW Description		
Switches the Development Error Detection and Notification ON or OFF.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context

Can	Can/CanGeneral		
BSW Parameter		**BSW Type**	
CanHardwareCancellation		EcucBooleanParamDef	
BSW Description			
Specifies if hardware cancellation shall be supported.ON or OFF			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	**BSW Context**		
Can	Can/CanGeneral		
BSW Parameter		**BSW Type**	
CanIdenticalIdCancellation		EcucBooleanParamDef	
BSW Description			
Enables/disables cancellation of pending PDUs with identical ID.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	**BSW Context**		
Can	Can/CanGeneral		
BSW Parameter		**BSW Type**	
CanIndex		EcucIntegerParamDef	
BSW Description			
Specifies the InstanceId of this module instance. If only one instance is present it shall have the Id 0.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	**BSW Context**	
Can	Can/CanGeneral	
BSW Parameter		**BSW Type**
CanLPduReceiveCalloutFunction		EcucFunctionNameDef
BSW Description		
This parameter defines the existence and the name of a callout function that is called after a successful reception of a received CAN Rx L-PDU. If this parameter is omitted no callout shall take place.		
M2 Template	**M2 Description**	
M2 Parameter		

Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**	
Can	Can/CanGeneral	
BSW Parameter		**BSW Type**
CanMainFunctionBusoffPeriod		EcucFloatParamDef
BSW Description		
This parameter describes the period for cyclic call to Can_MainFunction_Busoff. Unit is seconds.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
Can	Can/CanGeneral	
BSW Parameter		**BSW Type**
CanMainFunctionModePeriod		EcucFloatParamDef
BSW Description		
This parameter describes the period for cyclic call to Can_MainFunction_Mode. Unit is seconds.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
Can	Can/CanGeneral	
BSW Parameter		**BSW Type**
CanMainFunctionRWPeriods		EcucParamConfContainerDef
BSW Description		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
Can	Can/CanGeneral/CanMainFunctionRWPeriods	
BSW Parameter		**BSW Type**
CanMainFunctionReadPeriod		EcucFloatParamDef
BSW Description		

This parameter describes the period for cyclic call to Can_MainFunction_Read. Unit is seconds. Different poll-cycles will be configurable if more than one CanMainFunctionReadPeriod is configured. In this case multiple Can_MainFunction_Read() will be provided by the CAN Driver module.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**		
Can	Can/CanGeneral/CanMainFunctionRWPeriods		
BSW Parameter		**BSW Type**	
CanMainFunctionWritePeriod		EcucFloatParamDef	
BSW Description			
This parameter describes the period for cyclic call to Can_MainFunction_Write. Unit is seconds. Different poll-cycles will be configurable if more than one CanMainFunctionWritePeriod is configured. In this case multiple Can_MainFunction_Write() will be provided by the CAN Driver module.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	**BSW Context**		
Can	Can/CanGeneral		
BSW Parameter		**BSW Type**	
CanMainFunctionWakeupPeriod		EcucFloatParamDef	
BSW Description			
This parameter describes the period for cyclic call to Can_MainFunction_Wakeup. Unit is seconds.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	**BSW Context**		
Can	Can/CanGeneral		
BSW Parameter		**BSW Type**	
CanMultiplexedTransmission		EcucBooleanParamDef	
BSW Description			
Specifies if multiplexed transmission shall be supported.ON or OFF			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

— AUTOSAR CONFIDENTIAL —

BSW Module	**BSW Context**
Can	Can/CanGeneral
BSW Parameter	**BSW Type**
CanSupportTTCANRef	EcucReferenceDef
BSW Description	
The parameter refers to CanIfSupportTTCAN parameter in the CAN Interface Module configuration. The CanIfSupportTTCAN parameter defines whether TTCAN is supported.	
M2 Template	**M2 Description**
	Defines whether TTCAN is supported or not (reference to CanIfSupportTTCAN)
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**
Can	Can/CanGeneral
BSW Parameter	**BSW Type**
CanTimeoutDuration	EcucFloatParamDef
BSW Description	
Specifies the maximum time for blocking function until a timeout is detected. Unit is seconds.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**
Can	Can/CanGeneral
BSW Parameter	**BSW Type**
CanVersionInfoApi	EcucBooleanParamDef
BSW Description	
Switches the Can_GetVersionInfo() API ON or OFF.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

D.2 Can Interface Mapping

BSW Module	**BSW Context**
CanIf	CanIf
BSW Parameter	**BSW Type**
CanIfCtrlDrvCfg	EcucParamConfContainerDef
BSW Description	

Configuration parameters for all the underlying CAN Driver modules are aggregated under this container. For each CAN Driver module a seperate instance of this container has to be provided.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
CanIf	CanIf/CanIfCtrlDrvCfg	
BSW Parameter		**BSW Type**
CanIfCtrlCfg		EcucParamConfContainerDef
BSW Description		
This container contains the configuration (parameters) of an adressed CAN controller by an underlying CAN Driver module. This container is configurable per CAN controller.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
CanIf	CanIf/CanIfCtrlDrvCfg/CanIfCtrlCfg	
BSW Parameter		**BSW Type**
CanIfCtrlCanCtrlRef		EcucSymbolicNameReferenceDef
BSW Description		
This parameter references to the logical handle of the underlying CAN controller from the CAN Driver module to be served by the CAN Interface module. The following parameters of CanController config container shall be referenced by this link: CanControllerId, CanWakeupSourceRef Range: 0..max. number of underlying supported CAN controllers		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
CanIf	CanIf/CanIfCtrlDrvCfg/CanIfCtrlCfg	
BSW Parameter		**BSW Type**
CanIfCtrlId		EcucIntegerParamDef
BSW Description		
This parameter abstracts from the CAN Driver specific parameter Controller. Each controller of all connected CAN Driver modules shall be assigned to one specific ControllerId of the CanIf. Range: 0..number of configured controllers of all CAN Driver modules		
M2 Template	**M2 Description**	

M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	BSW Context	
CanIf	CanIf/CanIfCtrlDrvCfg/CanIfCtrlCfg	
BSW Parameter		**BSW Type**
CanIfCtrlWakeupSupport		EcucBooleanParamDef
BSW Description		
This parameter defines if a respective controller of the referenced CAN Driver modules is queriable for wake up events. True: Enabled False: Disabled		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
CanIf	CanIf/CanIfCtrlDrvCfg	
BSW Parameter		**BSW Type**
CanIfCtrlDrvInitHohConfigRef		EcucReferenceDef
BSW Description		
Reference to the Init Hoh Configuration		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
CanIf	CanIf/CanIfCtrlDrvCfg	
BSW Parameter		**BSW Type**
CanIfCtrlDrvNameRef		EcucReferenceDef
BSW Description		
CAN Interface Driver Reference. This reference can be used to get any information (Ex. Driver Name, Vendor ID) from the CAN driver. The CAN Driver name can be derived from the ShortName of the CAN driver module.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context
CanIf	CanIf/CanIfCtrlDrvCfg
BSW Parameter	**BSW Type**
CanIfCtrlDrvTxCancellation	EcucBooleanParamDef
BSW Description	
Selects whether transmit cancellation is supported and if the appropriate callback will be provided to the CAN Driver module. True: Enabled False: Disabled	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	BSW Context
CanIf	CanIf
BSW Parameter	**BSW Type**
CanIfDispatchCfg	EcucParamConfContainerDef
BSW Description	
Callback functions provided by upper layer modules of the CanIf. The callback functions defined in this container are common to all configured CAN Driver / CAN Transceiver Driver modules.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	BSW Context
CanIf	CanIf/CanIfDispatchCfg
BSW Parameter	**BSW Type**
CanIfDispatchUserCheckTrcvWakeFlagIndicationName	EcucFunctionNameDef
BSW Description	
This parameter defines the name of <User_ClearTrcvWufFlagIndication>. If CANIF_DISPATCH_USERCHECKTRCVWAKEFLAGINDICATION_UL equals CAN_SM the name of <User_CheckTrcvWakeFlagIndication> is fixed. If it equals CDD, the name is selectable. If CANIF_PUBLIC_PN_SUPPORT equals False, this parameter shall not be configurable.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**

BSW Module	BSW Context
CanIf	CanIf/CanIfDispatchCfg
BSW Parameter	**BSW Type**

CanIfDispatchUserCheckTrcvWakeFlagIndicationUL		EcucEnumerationParamDef
BSW Description		
This parameter defines the upper layer module to which the CheckTrcvWakeFlagIndication from the Driver modules have to be routed. If CANIF_PUBLIC_PN_SUPPORT equals False, this parameter shall not be configurable.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**

BSW Module	**BSW Context**	
CanIf	CanIf/CanIfDispatchCfg	
BSW Parameter		**BSW Type**
CanIfDispatchUserClearTrcvWufFlagIndicationName		EcucFunctionNameDef
BSW Description		
This parameter defines the name of <User_ClearTrcvWufFlagIndication>. If CANIF_DISPATCH_USERCLEARTRCVWUFFLAGINDICATION_UL equals CAN_SM the name of <User_ClearTrcvWufFlagIndication> is fixed. If it equals CDD, the name is selectable. If CANIF_PUBLIC_PN_SUPPORT equals False, this parameter shall not be configurable.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**

BSW Module	**BSW Context**	
CanIf	CanIf/CanIfDispatchCfg	
BSW Parameter		**BSW Type**
CanIfDispatchUserClearTrcvWufFlagIndicationUL		EcucEnumerationParamDef
BSW Description		
This parameter defines the upper layer module to which the ClearTrcvWufFlagIndication from the Driver modules have to be routed. If CANIF_PUBLIC_PN_SUPPORT equals False, this parameter shall not be configurable.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**

BSW Module	**BSW Context**	
CanIf	CanIf/CanIfDispatchCfg	
BSW Parameter		**BSW Type**
CanIfDispatchUserConfirmPnAvailabilityName		EcucFunctionNameDef
BSW Description		

This parameter defines the name of <User_ConfirmPnAvailability>. If CANIF_DISPATCH_USERCONFIRMPNAVAILABILITY_UL equals CAN_SM the name of <User_ConfirmPnAvailability> is fixed. If it equals CDD, the name is selectable. If CANIF_PUBLIC_PN_SUPPORT equals False, this parameter shall not be configurable.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**

BSW Module	**BSW Context**	
CanIf	CanIf/CanIfDispatchCfg	
BSW Parameter		**BSW Type**
CanIfDispatchUserConfirmPnAvailabilityUL		EcucEnumerationParamDef
BSW Description		
This parameter defines the upper layer module to which the ConfirmPnAvailability notification from the Driver modules have to be routed. If CANIF_PUBLIC_PN_SUPPORT equals False, this parameter shall not be configurable.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**

BSW Module	**BSW Context**	
CanIf	CanIf/CanIfDispatchCfg	
BSW Parameter		**BSW Type**
CanIfDispatchUserCtrlBusOffName		EcucFunctionNameDef
BSW Description		
This parameter defines the name of <User_ControllerBusOff>. This parameter depends on the parameter CANIF_USERCTRLBUSOFF_UL. If CANIF_USERCTRLBUSOFF_UL equals CAN_SM the name of <User_ControllerBusOff> is fixed. If CANIF_USERCTRLBUSOFF_UL equals CDD, the name of <User_ControllerBusOff> is selectable.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
CanIf	CanIf/CanIfDispatchCfg	
BSW Parameter		**BSW Type**
CanIfDispatchUserCtrlBusOffUL		EcucEnumerationParamDef
BSW Description		

This parameter defines the upper layer (UL) module to which the notifications of all ControllerBusOff events from the CAN Driver modules have to be routed via <User_ControllerBusOff>. There is no possibility to configure no upper layer (UL) module as the provider of <User_ControllerBusOff>.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**
CanIf	CanIf/CanIfDispatchCfg
BSW Parameter	**BSW Type**
CanIfDispatchUserCtrlModeIndicationName	EcucFunctionNameDef
BSW Description	
This parameter defines the name of <User_ControllerModeIndication>. This parameter depends on the parameter CANIF_USERCTRLMODEINDICATION_UL. If CANIF_USERCTRLMODEINDICATION_UL equals CAN_SM the name of <User_ControllerModeIndication> is fixed. If CANIF_USERCTRLMODEINDICATION_UL equals CDD, the name of <User_ControllerModeIndication> is selectable.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**
CanIf	CanIf/CanIfDispatchCfg
BSW Parameter	**BSW Type**
CanIfDispatchUserCtrlModeIndicationUL	EcucEnumerationParamDef
BSW Description	
This parameter defines the upper layer (UL) module to which the notifications of all ControllerTransition events from the CAN Driver modules have to be routed via <User_ControllerModeIndication>.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**
CanIf	CanIf/CanIfDispatchCfg
BSW Parameter	**BSW Type**
CanIfDispatchUserTrcvModeIndicationName	EcucFunctionNameDef
BSW Description	

This parameter defines the name of <User_TrcvModeIndication>. This parameter depends on the parameter CANIF_USERTRCVMODEINDICATION_UL. If CANIF_USERTRCVMODEINDICATION_UL equals CAN_SM the name of <User_TrcvModeIndication> is fixed. If CANIF_USERTRCVMODEINDICATION_UL equals CDD, the name of <User_TrcvModeIndication> is selectable.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**
CanIf	CanIf/CanIfDispatchCfg
BSW Parameter	**BSW Type**
CanIfDispatchUserTrcvModeIndicationUL	EcucEnumerationParamDef
BSW Description	
This parameter defines the upper layer (UL) module to which the notifications of all TransceiverTransition events from the CAN Transceiver Driver modules have to be routed via <User_TrcvModeIndication>. If no UL module is configured, no upper layer callback function will be called.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**
CanIf	CanIf/CanIfDispatchCfg
BSW Parameter	**BSW Type**
CanIfDispatchUserValidateWakeupEventName	EcucFunctionNameDef
BSW Description	
This parameter defines the name of <User_ValidateWakeupEvent>. This parameter depends on the parameter CANIF_USERVALIDATEWAKEUPEVENT_UL. CANIF_USERVALIDATEWAKEUPEVENT_UL equals ECUM the name of <User_ValidateWakeupEvent> is fixed. CANIF_USERVALIDATEWAKEUPEVENT_UL equals CDD, the name of <User_ValidateWakeupEvent> is selectable. If parameter CANIF_WAKEUP_CHECK_VALIDATION_API is disabled, no <User_ValidateWakeupEvent> API can be configured.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**
CanIf	CanIf/CanIfDispatchCfg
BSW Parameter	**BSW Type**
CanIfDispatchUserValidateWakeupEventUL	EcucEnumerationParamDef

<table>
<tr><th colspan="3">BSW Description</th></tr>
<tr><td colspan="3">This parameter defines the upper layer (UL) module to which the notifications about positive former requested wake up sources have to be routed via <User_ValidateWakeupEvent>. If parameter CANIF_WAKEUP_CHECK_VALIDATION_API is disabled, this parameter cannot be configured.</td></tr>
<tr><th>M2 Template</th><th colspan="2">M2 Description</th></tr>
<tr><td></td><td colspan="2"></td></tr>
<tr><th colspan="3">M2 Parameter</th></tr>
<tr><td colspan="3"></td></tr>
<tr><th colspan="2">Mapping Rule</th><th>Mapping Type</th></tr>
<tr><td colspan="2"></td><td>local</td></tr>
</table>

<table>
<tr><th>BSW Module</th><th colspan="2">BSW Context</th></tr>
<tr><td>CanIf</td><td colspan="2">CanIf</td></tr>
<tr><th colspan="2">BSW Parameter</th><th>BSW Type</th></tr>
<tr><td colspan="2">CanIfInitCfg</td><td>EcucParamConfContainerDef</td></tr>
<tr><th colspan="3">BSW Description</th></tr>
<tr><td colspan="3">This container contains the init parameters of the CAN Interface.

At least one (if only on CanIf with one possible Configuration), but multiple (CanIf with different Configurations) instances of this container are possible.</td></tr>
<tr><th>M2 Template</th><th colspan="2">M2 Description</th></tr>
<tr><td></td><td colspan="2"></td></tr>
<tr><th colspan="3">M2 Parameter</th></tr>
<tr><td colspan="3"></td></tr>
<tr><th colspan="2">Mapping Rule</th><th>Mapping Type</th></tr>
<tr><td colspan="2"></td><td>local</td></tr>
</table>

<table>
<tr><th>BSW Module</th><th colspan="2">BSW Context</th></tr>
<tr><td>CanIf</td><td colspan="2">CanIf/CanIfInitCfg</td></tr>
<tr><th colspan="2">BSW Parameter</th><th>BSW Type</th></tr>
<tr><td colspan="2">CanIfBufferCfg</td><td>EcucParamConfContainerDef</td></tr>
<tr><th colspan="3">BSW Description</th></tr>
<tr><td colspan="3">This container contains the Txbuffer configuration. Multiple buffers with different sizes could be configured. If CanIfBufferSize (CANIF834_Conf) equals 0, the CanIf Tx L-PDU only refers via this CanIfBufferCfg the
corresponding CanIfHthCfg.</td></tr>
<tr><th>M2 Template</th><th colspan="2">M2 Description</th></tr>
<tr><td></td><td colspan="2"></td></tr>
<tr><th colspan="3">M2 Parameter</th></tr>
<tr><td colspan="3"></td></tr>
<tr><th colspan="2">Mapping Rule</th><th>Mapping Type</th></tr>
<tr><td colspan="2"></td><td>local</td></tr>
</table>

<table>
<tr><th>BSW Module</th><th colspan="2">BSW Context</th></tr>
<tr><td>CanIf</td><td colspan="2">CanIf/CanIfInitCfg/CanIfBufferCfg</td></tr>
<tr><th colspan="2">BSW Parameter</th><th>BSW Type</th></tr>
<tr><td colspan="2">CanIfBufferHthRef</td><td>EcucReferenceDef</td></tr>
<tr><th colspan="3">BSW Description</th></tr>
<tr><td colspan="3">Reference to HTH, that defines the hardware object or the pool of hardware objects configured for transmission. All the CanIf Tx L-PDUs refer via the CanIfBufferCfg and this parameter to the HTHs if TxBuffering is enabled, or not.</td></tr>
</table>

M2 Template	M2 Description	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
CanIf	CanIf/CanIfInitCfg/CanIfBufferCfg	
BSW Parameter		**BSW Type**
CanIfBufferSize		EcucIntegerParamDef
BSW Description		
This parameter defines the number of CanIf Tx L-PDUs which can be buffered in one Txbuffer. If this value equals 0, the CanIf does not perform Txbuffering for the CanIf Tx L-PDUs which are assigned to this Txbuffer. If CanIfPublicTxBuffering equals False, this parameter equals 0 for all TxBuffer. If the CanHandleType of the referred HTH equals FULL, this parameter equals 0 for this TxBuffer.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
CanIf	CanIf/CanIfInitCfg	
BSW Parameter		**BSW Type**
CanIfInitCfgSet		EcucStringParamDef
BSW Description		
Selects the CAN Interface specific configuration setup. This type of the external data structure shall contain the post build initialization data for the CAN Interface for all underlying CAN Dirvers. constant to CanIf_ConfigType		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
CanIf	CanIf/CanIfInitCfg	
BSW Parameter		**BSW Type**
CanIfInitHohCfg		EcucParamConfContainerDef
BSW Description		
This container contains the references to the configuration setup of each underlying CAN Driver.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context		
CanIf	CanIf/CanIfInitCfg/CanIfInitHohCfg		
BSW Parameter		**BSW Type**	
CanIfHrhCfg		EcucParamConfContainerDef	
BSW Description			
This container contains configuration parameters for each hardware receive object (HRH).			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	BSW Context		
CanIf	CanIf/CanIfInitCfg/CanIfInitHohCfg/CanIfHrhCfg		
BSW Parameter		**BSW Type**	
CanIfHrhCanCtrlIdRef		EcucReferenceDef	
BSW Description			
Reference to controller Id to which the HRH belongs to. A controller can contain one or more HRHs.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	BSW Context		
CanIf	CanIf/CanIfInitCfg/CanIfInitHohCfg/CanIfHrhCfg		
BSW Parameter		**BSW Type**	
CanIfHrhCanHandleTypeRef		EcucSymbolicNameReferenceDef	
BSW Description			
The parameter refers to a particular HRH object in the CAN Driver Module configuration. The type of the HRH can either be Full-CAN or Basic-CAN. The type of HRHs is defined in the CAN Driver Module and hence it is derived from CAN Driver Configuration of a Hardware Object. If BasicCAN is configured, software filtering is enabled. Please note that this reference is deprecated and is kept only for backward compatibility reasons. CanIfHthIdSymRef shall be used instead to get the CanHandleType and CanObjectId of CAN Driver. In the next major release this reference will be deleted.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	BSW Context	
CanIf	CanIf/CanIfInitCfg/CanIfInitHohCfg/CanIfHrhCfg	
BSW Parameter		**BSW Type**
CanIfHrhIdSymRef		EcucSymbolicNameReferenceDef

BSW Description	
The parameter refers to a particular HRH object in the CanDrv configuration (see CanHardwareObject CAN324_Conf). The CanIf receives the following information of the CanDrv module by this reference: - CanHandleType (see CAN323_Conf) - CanObjectId (see CAN326_Conf)	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	BSW Context
CanIf	CanIf/CanIfInitCfg/CanIfInitHohCfg/CanIfHrhCfg
BSW Parameter	**BSW Type**
CanIfHrhRangeCfg	EcucParamConfContainerDef
BSW Description	
Defines the parameters required for configurating multiple CANID ranges for a given same HRH.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	BSW Context
CanIf	CanIf/CanIfInitCfg/CanIfInitHohCfg/CanIfHrhCfg/CanIfHrhRangeCfg
BSW Parameter	**BSW Type**
CanIfHrhRangeRxPduLowerCanId	EcucIntegerParamDef
BSW Description	
Lower CAN Identifier of a receive CAN L-PDU for identifier range definition, in which all CAN Ids shall pass the software filtering.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	BSW Context
CanIf	CanIf/CanIfInitCfg/CanIfInitHohCfg/CanIfHrhCfg/CanIfHrhRangeCfg
BSW Parameter	**BSW Type**
CanIfHrhRangeRxPduRangeCanIdType	EcucEnumerationParamDef
BSW Description	
Specifies whether a configured Range of CAN Ids shall only consider standard CAN Ids or extended CAN Ids.	
M2 Template	**M2 Description**

M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	BSW Context	
CanIf	CanIf/CanIfInitCfg/CanIfInitHohCfg/CanIfHrhCfg/CanIfHrhRangeCfg	
BSW Parameter		**BSW Type**
CanIfHrhRangeRxPduUpperCanId		EcucIntegerParamDef
BSW Description		
Upper CAN Identifier of a receive CAN L-PDU for identifier range definition, in which all CAN Ids shall pass the software filtering.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
CanIf	CanIf/CanIfInitCfg/CanIfInitHohCfg/CanIfHrhCfg	
BSW Parameter		**BSW Type**
CanIfHrhSoftwareFilter		EcucBooleanParamDef
BSW Description		
Selects the hardware receive objects by using the HRH range/list from CAN Driver configuration to define, for which HRH a software filtering has to be performed at during receive processing. True: Software filtering is enabled False: Software filtering is enabled		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
CanIf	CanIf/CanIfInitCfg/CanIfInitHohCfg	
BSW Parameter		**BSW Type**
CanIfHthCfg		EcucParamConfContainerDef
BSW Description		
This container contains parameters related to each HTH.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
CanIf	CanIf/CanIfInitCfg/CanIfInitHohCfg/CanIfHthCfg	
BSW Parameter		**BSW Type**
CanIfHthCanCtrlIdRef		EcucReferenceDef
BSW Description		
Reference to controller Id to which the HTH belongs to. A controller can contain one or more HTHs.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
CanIf	CanIf/CanIfInitCfg/CanIfInitHohCfg/CanIfHthCfg	
BSW Parameter		**BSW Type**
CanIfHthCanHandleTypeRef		EcucSymbolicNameReferenceDef
BSW Description		
The parameter refers to a particular HTH object in the CAN Driver Module configuration. The type of the HTH can either be Full-CAN or Basic-CAN. The type of HTHs is defined in the CAN Driver Module and hence it is derived from CAN Driver Configuration of a Hardware Object. Please note that this reference is deprecated and is kept only for backward compatibility reasons. CanIfHthIdSymRef shall be used instead to get the CanHandleType and CanObjectId of CAN Driver. In the next major release this reference will be deleted.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
CanIf	CanIf/CanIfInitCfg/CanIfInitHohCfg/CanIfHthCfg	
BSW Parameter		**BSW Type**
CanIfHthIdSymRef		EcucSymbolicNameReferenceDef
BSW Description		
The parameter refers to a particular HTH object in the CanDrv configuration (see CanHardwareObject CAN324_Conf). The CanIf receives the following information of the CanDrv module by this reference: - CanHandleType (see CAN323_Conf) - CanObjectId (see CAN326_Conf)		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context

CanIf	CanIf/CanIfInitCfg/CanIfInitHohCfg	
BSW Parameter		**BSW Type**
CanIfInitRefCfgSet		EcucReferenceDef
BSW Description		
Selects the CAN Interface specific configuration setup. This type of external data structure shall contain the post build initialization data for the CAN Interface for all underlying CAN Drivers.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
CanIf	CanIf/CanIfInitCfg	
BSW Parameter		**BSW Type**
CanIfRxPduCfg		EcucParamConfContainerDef
BSW Description		
This container contains the configuration (parameters) of each receive CAN L-PDU. The SHORT-NAME of "CanIfRxPduConfig" container itself represents the symolic name of Receive L-PDU.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
CanIf	CanIf/CanIfInitCfg/CanIfRxPduCfg	
BSW Parameter		**BSW Type**
CanIfRxPduBswSchExclAreaIdRef		EcucReferenceDef
BSW Description		
Reference to an exclusive area Id defined within the BSW Scheduler.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
CanIf	CanIf/CanIfInitCfg/CanIfRxPduCfg	
BSW Parameter		**BSW Type**
CanIfRxPduCanId		EcucIntegerParamDef
BSW Description		

CAN Identifier of Receive CAN L-PDUs used by the CAN Interface. Exa: Software Filtering. This parameter is used if exactly one Can Identifier is assigned to the Pdu. If a range is assigned then the CanIfRxPduCanIdRange parameter shall be used. Range: 11 Bit For Standard CAN Identifier ... 29 Bit For Extended CAN identifier		
M2 Template	**M2 Description**	
SystemTemplate	To describe a frames identifier on the communication system, usualy with a fixed identifierValue.	
M2 Parameter		
Fibex::Fibex4Can::CanCommunication::CanFrameTriggering.identifier		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**		
CanIf	CanIf/CanIfInitCfg/CanIfRxPduCfg		
BSW Parameter		**BSW Type**	
CanIfRxPduCanIdRange		EcucParamConfContainerDef	
BSW Description			
Optional container that allows to map a range of CAN Ids to one PduId.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	**BSW Context**		
CanIf	CanIf/CanIfInitCfg/CanIfRxPduCfg/CanIfRxPduCanIdRange		
BSW Parameter		**BSW Type**	
CanIfRxPduCanIdRangeLowerCanId		EcucIntegerParamDef	
BSW Description			
Lower CAN Identifier of a receive CAN L-PDU for identifier range definition, in which all CAN Ids are mapped to one PduId.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	**BSW Context**	
CanIf	CanIf/CanIfInitCfg/CanIfRxPduCfg/CanIfRxPduCanIdRange	
BSW Parameter		**BSW Type**
CanIfRxPduCanIdRangeUpperCanId		EcucIntegerParamDef
BSW Description		
Upper CAN Identifier of a receive CAN L-PDU for identifier range definition, in which all CAN Ids are mapped to one PduId.		
M2 Template	**M2 Description**	
M2 Parameter		

Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**	
CanIf	CanIf/CanIfInitCfg/CanIfRxPduCfg	
BSW Parameter		**BSW Type**
CanIfRxPduCanIdType		EcucEnumerationParamDef
BSW Description		
CAN Identifier of receive CAN L-PDUs used by the CAN Driver for CAN L-PDU reception.		
M2 Template	**M2 Description**	
SystemTemplate	... two types of frame formats. The standard frame format uses 11-bit identifiers ... the extended frame format allows 29-bit identifiers ...	
M2 Parameter		
Fibex::Fibex4Can::CanCommunication::CanFrameTriggering.canAddressingMode		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
CanIf	CanIf/CanIfInitCfg/CanIfRxPduCfg	
BSW Parameter		**BSW Type**
CanIfRxPduDlc		EcucIntegerParamDef
BSW Description		
Data Length code of received CAN L-PDUs used by the CAN Interface. Exa: DLC check. The data area size of a CAN L-PDU can have a range from 0 to 8 bytes.		
M2 Template	**M2 Description**	
SystemTemplate	The used length (in bytes) of the referencing frame. Should not be confused with a static byte length reserved for each frame by some platforms (e.g. FlexRay).	
M2 Parameter		
Fibex::FibexCore::CoreCommunication::Frame.frameLength		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
CanIf	CanIf/CanIfInitCfg/CanIfRxPduCfg	
BSW Parameter		**BSW Type**
CanIfRxPduHrhIdRef		EcucReferenceDef
BSW Description		
The HRH to which Rx L-PDU belongs to, is referred through this parameter.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**
CanIf	CanIf/CanIfInitCfg/CanIfRxPduCfg

BSW Parameter		BSW Type	
CanIfRxPduId		EcucIntegerParamDef	
BSW Description			
ECU wide unique, symbolic handle for receive CAN L-PDU. The CanIfRxPduId is configurable at pre-compile and post-built time. It shall fulfill ANSI/AUTOSAR definitions for constant defines. Range: 0..max. number of defined CanRxPduIds			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	BSW Context		
CanIf	CanIf/CanIfInitCfg/CanIfRxPduCfg		
BSW Parameter		**BSW Type**	
CanIfRxPduReadData		EcucBooleanParamDef	
BSW Description			
Enables and disables the Rx buffering for reading of received L-PDU data. True: Enabled False: Disabled			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	BSW Context		
CanIf	CanIf/CanIfInitCfg/CanIfRxPduCfg		
BSW Parameter		**BSW Type**	
CanIfRxPduReadNotifyStatus		EcucBooleanParamDef	
BSW Description			
Enables and disables receive indication for each receive CAN L-PDU for reading its notification status. True: Enabled False: Disabled			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	BSW Context	
CanIf	CanIf/CanIfInitCfg/CanIfRxPduCfg	
BSW Parameter		**BSW Type**
CanIfRxPduRef		EcucReferenceDef

— AUTOSAR CONFIDENTIAL —

BSW Description	
Reference to the "global" Pdu structure to allow harmonization of handle IDs in the COM-Stack.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**
CanIf	CanIf/CanIfInitCfg/CanIfRxPduCfg
BSW Parameter	**BSW Type**
CanIfRxPduUserRxIndicationName	EcucFunctionNameDef
BSW Description	
This parameter defines the name of the <User_RxIndication>. This parameter depends on the parameter CANIF_RXPDU_USERRXINDICATION_UL. If CANIF_RXPDU_USERRXINDICATION_UL equals CAN_TP, CAN_NM, PDUR, XCP or J1939TP, the name of the <User_RxIndication> is fixed. If CANIF_RXPDU_USERRXINDICATION_UL equals CDD, the name of the <User_RxIndication> is selectable.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**
CanIf	CanIf/CanIfInitCfg/CanIfRxPduCfg
BSW Parameter	**BSW Type**
CanIfRxPduUserRxIndicationUL	EcucEnumerationParamDef
BSW Description	
This parameter defines the upper layer (UL) module to which the indication of the successfully received CANRXPDUID has to be routed via <User_RxIndication>. This <User_RxIndication> has to be invoked when the indication of the configured CANRXPDUID will be received by an Rx indication event from the CAN Driver module. If no upper layer (UL) module is configured, no <User_RxIndication> has to be called in case of an Rx indication event of the CANRXPDUID from the CAN Driver module.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**
CanIf	CanIf/CanIfInitCfg/CanIfRxPduCfg
BSW Parameter	**BSW Type**
CanIfTTRxFrameTriggering	EcucParamConfContainerDef
BSW Description	

This container is only included and valid if TTCAN Interface SWS is used and TTCAN is enabled. Frame trigger for TTCAN reception. CanIfTTRxFrameTriggering is only included, if the controller supports TTCAN and a joblist is used for reception.		
M2 Template	**M2 Description**	
SystemTemplate	CAN specific attributes to the FrameTriggering	
M2 Parameter		
CanCommunication::CanFrameTriggering		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
CanIf	CanIf/CanIfInitCfg/CanIfRxPduCfg/CanIfTTRxFrameTriggering	
BSW Parameter		**BSW Type**
CanIfTTRxHwObjectTriggerIdRef		EcucReferenceDef
BSW Description		
This parameter refers to a particular TTCAN hardware receive object Trigger of a hardware object in the TTCAN Driver Module, which is referred via plain CAN parameter CANIF_HRH_HANDLETYPE_REF. This parameter is only configurable if a joblist is enabled by parameter CanIfTTJobList.		
M2 Template	**M2 Description**	
	reference to a specific trigger defined for a HRH	
M2 Parameter		
Mapping Rule		**Mapping Type**

BSW Module	**BSW Context**	
CanIf	CanIf/CanIfInitCfg/CanIfRxPduCfg/CanIfTTRxFrameTriggering	
BSW Parameter		**BSW Type**
CanTTRxJoblistTimeMark		EcucIntegerParamDef
BSW Description		
Defines the point in time, when the joblist execution funciton (JLEF) shall be called for the referenced rx trigger. Value is given in cycle time. This parameter is only configurable if a joblist is enabled by parameter CanIfTTJobList.		
M2 Template	**M2 Description**	
	Time mark for calling the joblist (for message processing)	
M2 Parameter		
Mapping Rule		**Mapping Type**

BSW Module	**BSW Context**	
CanIf	CanIf/CanIfInitCfg	
BSW Parameter		**BSW Type**
CanIfTxPduCfg		EcucParamConfContainerDef
BSW Description		

This container contains the configuration (parameters) of a transmit CAN L-PDU. It has to be configured as often as a transmit CAN L-PDU is needed. The SHORT-NAME of "CanIfTxPduConfig" container represents the symolic name of Transmit L-PDU.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**		
CanIf	CanIf/CanIfInitCfg/CanIfTxPduCfg		
BSW Parameter		**BSW Type**	
CanIfTTTxFrameTriggering		EcucParamConfContainerDef	
BSW Description			
This container is only included and valid if TTCAN Interface SWS is used and TTCAN is enabled. Frame trigger for TTCAN transmission. CanIfTTTxFrameTriggering is only included, if the controller supports TTCAN and a joblist is used.			
M2 Template	**M2 Description**		
SystemTemplate	CAN specific attributes to the FrameTriggering		
M2 Parameter			
CanCommunication::CanFrameTriggering			
Mapping Rule			**Mapping Type**
1:1 mapping			full

BSW Module	**BSW Context**		
CanIf	CanIf/CanIfInitCfg/CanIfTxPduCfg/CanIfTTTxFrameTriggering		
BSW Parameter		**BSW Type**	
CanIfTTTxHwObjectTriggerIdRef		EcucReferenceDef	
BSW Description			
This parameter refers to a particular TTCAN hardware transmit object Trigger of a hardware object in the TTCAN Driver Module, which is referred via plain CAN parameter CANIF_HTH_HANDLETYPE_REF. This parameter is only configurable if a joblist is enabled by parameter CanIfTTJobList.			
M2 Template	**M2 Description**		
	reference to a specific trigger defined for a HTH		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	**BSW Context**	
CanIf	CanIf/CanIfInitCfg/CanIfTxPduCfg/CanIfTTTxFrameTriggering	
BSW Parameter		**BSW Type**
CanIfTTTxJoblistTimeMark		EcucIntegerParamDef
BSW Description		

Defines the point in time, when the joblist execution funciton (JLEF) shall be called for the referenced tx frame trigger. Value is given in cycle time. This parameter is only configurable if a joblist is enabled by parameter CanIfTTJobList.		
M2 Template	**M2 Description**	
	Time mark for calling the joblist (for message processing)	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
CanIf	CanIf/CanIfInitCfg/CanIfTxPduCfg	
BSW Parameter		**BSW Type**
CanIfTxPduBswSchExclAreaIdRef		EcucReferenceDef
BSW Description		
Reference to an exclusive area Id defined within the BSW Scheduler.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
CanIf	CanIf/CanIfInitCfg/CanIfTxPduCfg	
BSW Parameter		**BSW Type**
CanIfTxPduBufferRef		EcucReferenceDef
BSW Description		
Configurable reference to a CanIf buffer configuration.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
CanIf	CanIf/CanIfInitCfg/CanIfTxPduCfg	
BSW Parameter		**BSW Type**
CanIfTxPduCanId		EcucIntegerParamDef
BSW Description		
CAN Identifier of transmit CAN L-PDUs used by the CAN Driver for CAN L-PDU transmission. Range: 11 Bit For Standard CAN Identifier ... 29 Bit For Extended CAN identifier		
M2 Template	**M2 Description**	
SystemTemplate	To describe a frames identifier on the communication system, usualy with a fixed identifierValue.	
M2 Parameter		
Fibex::Fibex4Can::CanCommunication::CanFrameTriggering.identifier		
Mapping Rule		**Mapping Type**

1:1 mapping	full

BSW Module	**BSW Context**	
CanIf	CanIf/CanIfInitCfg/CanIfTxPduCfg	
BSW Parameter		**BSW Type**
CanIfTxPduCanIdType		EcucEnumerationParamDef
BSW Description		
Type of CAN Identifier of the transmit CAN L-PDU used by the CAN Driver module for CAN L-PDU transmission.		
M2 Template	**M2 Description**	
SystemTemplate	... two types of frame formats. The standard frame format uses 11-bit identifiers ... the extended frame format allows 29-bit identifiers ...	
M2 Parameter		
Fibex::Fibex4Can::CanCommunication::CanFrameTriggering.canAddressingMode		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
CanIf	CanIf/CanIfInitCfg/CanIfTxPduCfg	
BSW Parameter		**BSW Type**
CanIfTxPduDlc		EcucIntegerParamDef
BSW Description		
Data length code (in bytes) of transmit CAN L-PDUs used by the CAN Driver for CAN L-PDU transmission. The data area size of a CAN L-Pdu can have a range from 0 to 8 bytes.		
M2 Template	**M2 Description**	
SystemTemplate	The used length (in bytes) of the referencing frame. Should not be confused with a static byte length reserved for each frame by some platforms (e.g. FlexRay).	
M2 Parameter		
Fibex::FibexCore::CoreCommunication::Frame.frameLength		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
CanIf	CanIf/CanIfInitCfg/CanIfTxPduCfg	
BSW Parameter		**BSW Type**
CanIfTxPduId		EcucIntegerParamDef
BSW Description		
ECU wide unique, symbolic handle for transmit CAN L-PDU. The CanIfTxPduId is configurable at pre-compile and post-built time. Range: 0..max. number of CantTxPduIds		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context		
CanIf	CanIf/CanIfInitCfg/CanIfTxPduCfg		
BSW Parameter		**BSW Type**	
CanIfTxPduPnFilterPdu		EcucBooleanParamDef	
BSW Description			
If CanIfPublicPnFilterSupport is enabled, by this parameter PDUs could be configured which will pass the CanIfPnFilter. If there is no CanIfTxPduPnFilterPdu configured per controller, the corresponding controller applies no CanIfPnFilter.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	BSW Context		
CanIf	CanIf/CanIfInitCfg/CanIfTxPduCfg		
BSW Parameter		**BSW Type**	
CanIfTxPduReadNotifyStatus		EcucBooleanParamDef	
BSW Description			
Enables and disables transmit confirmation for each transmit CAN L-PDU for reading its notification status. True: Enabled False: Disabled			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	BSW Context		
CanIf	CanIf/CanIfInitCfg/CanIfTxPduCfg		
BSW Parameter		**BSW Type**	
CanIfTxPduRef		EcucReferenceDef	
BSW Description			
Reference to the "global" Pdu structure to allow harmonization of handle IDs in the COM-Stack.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	BSW Context	
CanIf	CanIf/CanIfInitCfg/CanIfTxPduCfg	
BSW Parameter		**BSW Type**
CanIfTxPduType		EcucEnumerationParamDef

BSW Description		
Defines the type of each transmit CAN L-PDU.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context		
CanIf	CanIf/CanIfInitCfg/CanIfTxPduCfg		
BSW Parameter		**BSW Type**	
CanIfTxPduUserTxConfirmationName		EcucFunctionNameDef	
BSW Description			
This parameter defines the name of the <User_TxConfirmation>. This parameter depends on the parameter CANIF_TXPDU_USERTXCONFIRMATION_UL. If CANIF_TXPDU_USERTXCONFIRMATION_UL equals CAN_TP, CAN_NM, PDUR, XCP or J1939TP, the name of the <User_TxConfirmation> is fixed. If CANIF_TXPDU_USERTXCONFIRMATION_UL equals CDD, the name of the <User_TxConfirmation> is selectable.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	BSW Context		
CanIf	CanIf/CanIfInitCfg/CanIfTxPduCfg		
BSW Parameter		**BSW Type**	
CanIfTxPduUserTxConfirmationUL		EcucEnumerationParamDef	
BSW Description			
This parameter defines the upper layer (UL) module to which the confirmation of the successfully transmitted CANTXPDUID has to be routed via the <User_TxConfirmation>. This <User_TxConfirmation> has to be invoked when the confirmation of the configured CANTXPDUID will be received by a Tx confirmation event from the CAN Driver module. If no upper layer (UL) module is configured, no <User_TxConfirmation> has to be called in case of a Tx confirmation event of the CANTXPDUID from the CAN Driver module.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	BSW Context	
CanIf	CanIf	
BSW Parameter		**BSW Type**
CanIfPrivateCfg		EcucParamConfContainerDef
BSW Description		
This container contains the private configuration (parameters) of the CAN Interface.		

M2 Template	M2 Description	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context		
CanIf	CanIf/CanIfPrivateCfg		
BSW Parameter		**BSW Type**	
CanIfPrivateDlcCheck		EcucBooleanParamDef	
BSW Description			
Selects whether the DLC check is supported. True: Enabled False: Disabled			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	BSW Context		
CanIf	CanIf/CanIfPrivateCfg		
BSW Parameter		**BSW Type**	
CanIfPrivateSoftwareFilterType		EcucEnumerationParamDef	
BSW Description			
Selects the desired software filter mechanism for reception only. Each implemented software filtering method is identified by this enumeration number. Range: Types implemented software filtering methods			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	BSW Context	
CanIf	CanIf/CanIfPrivateCfg	
BSW Parameter		**BSW Type**
CanIfSupportTTCAN		EcucBooleanParamDef
BSW Description		
Defines whether TTCAN is supported. TRUE: TTCAN is supported. FALSE: TTCAN is not supported, only normal CAN communication is possible.		
M2 Template	**M2 Description**	
	Defines whether TTCAN is supported or not	
M2 Parameter		

Mapping Rule	**Mapping Type**

BSW Module	**BSW Context**	
CanIf	CanIf/CanIfPrivateCfg	
BSW Parameter		**BSW Type**
CanIfTTGeneral		EcucParamConfContainerDef
BSW Description		
This container is only included and valid if TTCAN Interface SWS is used and TTCAN is enabled. This container contains the parameters, which define if and in which way TTCAN is supported. CanIfTTGeneral is only included, if the controller supports TTCAN.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
CanIf	CanIf/CanIfPrivateCfg/CanIfTTGeneral	
BSW Parameter		**BSW Type**
CanIfTTJoblist		EcucBooleanParamDef
BSW Description		
Defines whether TTCAN is processed via a joblist. TRUE: Joblist is used. FALSE: No joblist is used. This parameter is only configurable if TTCAN is enabled by parameter CanIfSupportTTCAN.		
M2 Template	**M2 Description**	
	Defines whehter a joblist is used or not	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
CanIf	CanIf/CanIfPrivateCfg/CanIfTTGeneral	
BSW Parameter		**BSW Type**
CanIfTTMaxIsrDelay		EcucIntegerParamDef
BSW Description		
Defines the maximum delay for the execution of the joblist execution function JLEF. This parameter is only configurable if a joblist is enabled by parameter CanIfTTJobList.		
M2 Template	**M2 Description**	
	Defines the maximum delay for the job list execution	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context
CanIf	CanIf
BSW Parameter	**BSW Type**
CanIfPublicCfg	EcucParamConfContainerDef
BSW Description	
This container contains the public configuration (parameters) of the CAN Interface.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	BSW Context
CanIf	CanIf/CanIfPublicCfg
BSW Parameter	**BSW Type**
CanIfPublicCancelTransmitSupport	EcucBooleanParamDef
BSW Description	
Configuration parameter to enable/disable dummy API for upper layer modules which allows to request the cancellation of an I-PDU.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	BSW Context
CanIf	CanIf/CanIfPublicCfg
BSW Parameter	**BSW Type**
CanIfPublicCddHeaderFile	EcucStringParamDef
BSW Description	
Defines header files for callback functions which shall be included in case of CDDs. Range of characters is 1.. 32.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	BSW Context
CanIf	CanIf/CanIfPublicCfg
BSW Parameter	**BSW Type**
CanIfPublicChangeBaudrateSupport	EcucBooleanParamDef
BSW Description	
Configuration parameter to enable/disable the API to change the baudrate of a CAN controller. True: Enabled False: Disabled	
M2 Template	**M2 Description**

M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**	
CanIf	CanIf/CanIfPublicCfg	
BSW Parameter		**BSW Type**
CanIfPublicDevErrorDetect		EcucBooleanParamDef
BSW Description		
Enables and disables the development error detection and notification mechanism. True: Enabled False: Disabled		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
CanIf	CanIf/CanIfPublicCfg	
BSW Parameter		**BSW Type**
CanIfPublicHandleTypeEnum		EcucEnumerationParamDef
BSW Description		
This parameter is used to configure the Can_HwHandleType. The Can_HwHandleType represents the hardware object handles of a CAN hardware unit. For CAN hardware units with more than 255 HW objects the extended range shall be used (UINT16).		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
CanIf	CanIf/CanIfPublicCfg	
BSW Parameter		**BSW Type**
CanIfPublicMultipleDrvSupport		EcucBooleanParamDef
BSW Description		
Selects support for multiple CAN Drivers. True: Enabled False: Disabled		
M2 Template	**M2 Description**	
M2 Parameter		

Mapping Rule	Mapping Type
	local

BSW Module	BSW Context	
CanIf	CanIf/CanIfPublicCfg	
BSW Parameter		**BSW Type**
CanIfPublicNumberOfCanHwUnits		EcucIntegerParamDef
BSW Description		
Number of served CAN hardware units.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
CanIf	CanIf/CanIfPublicCfg	
BSW Parameter		**BSW Type**
CanIfPublicPnSupport		EcucBooleanParamDef
BSW Description		
Selects support of Partial Network features in CanIf. True: Enabled False: Disabled		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
CanIf	CanIf/CanIfPublicCfg	
BSW Parameter		**BSW Type**
CanIfPublicReadRxPduDataApi		EcucBooleanParamDef
BSW Description		
Enables / Disables the API CanIf_ReadRxPduData() for reading received L-PDU data. True: Enabled False: Disabled		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context
CanIf	CanIf/CanIfPublicCfg

BSW Parameter		BSW Type	
CanIfPublicReadRxPduNotifyStatusApi		EcucBooleanParamDef	
BSW Description			
Enables and disables the API for reading the received L-PDU data. True: Enabled False: Disabled			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	BSW Context		
CanIf	CanIf/CanIfPublicCfg		
BSW Parameter		**BSW Type**	
CanIfPublicReadTxPduNotifyStatusApi		EcucBooleanParamDef	
BSW Description			
Enables and disables the API for reading the notification status of transmit and receive L-PDUs. True: Enabled False: Disabled			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	BSW Context		
CanIf	CanIf/CanIfPublicCfg		
BSW Parameter		**BSW Type**	
CanIfPublicSetDynamicTxIdApi		EcucBooleanParamDef	
BSW Description			
Enables and disables the API for reconfiguration of the CAN Identifier for each Transmit L-PDU. True: Enabled False: Disabled			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	BSW Context	
CanIf	CanIf/CanIfPublicCfg	
BSW Parameter		**BSW Type**
CanIfPublicTxBuffering		EcucBooleanParamDef
BSW Description		

Enables and disables the buffering of transmit L-PDUs (rejected by the CanDrv) within the CAN Interface module. True: Enabled False: Disabled		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
CanIf	CanIf/CanIfPublicCfg	
BSW Parameter		**BSW Type**
CanIfPublicTxConfirmPollingSupport		EcucBooleanParamDef
BSW Description		
Configuration parameter to enable/disable the API to poll for Tx Confirmation state.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
CanIf	CanIf/CanIfPublicCfg	
BSW Parameter		**BSW Type**
CanIfPublicVersionInfoApi		EcucBooleanParamDef
BSW Description		
Enables and disables the API for reading the version information about the CAN Interface. True: Enabled False: Disabled		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
CanIf	CanIf/CanIfPublicCfg	
BSW Parameter		**BSW Type**
CanIfPublicWakeupCheckValidByNM		EcucBooleanParamDef
BSW Description		

— AUTOSAR CONFIDENTIAL —

If enabled, only NM messages shall validate a detected wake-up event (see CANIF722) at the corresponding wake-up source in the CanIf. If disabled, all messages shall validate such a wake-up event. This parameter depends on CANIF_PUBLIC_WAKEUP_CHECK_VALID_API and shall only be configurable, if it is enabled. True: Enabled False: Disabled		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**		
CanIf	CanIf/CanIfPublicCfg		
BSW Parameter		**BSW Type**	
CanIfPublicWakeupCheckValidSupport		EcucBooleanParamDef	
BSW Description			
Selects support for wake up validation True: Enabled False: Disabled			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	**BSW Context**		
CanIf	CanIf		
BSW Parameter		**BSW Type**	
CanIfTrcvDrvCfg		EcucParamConfContainerDef	
BSW Description			
This container contains the configuration (parameters) of all addressed CAN transceivers by each underlying CAN Transceiver Driver module. For each CAN transceiver Driver a seperate instance of this container shall be provided.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	**BSW Context**	
CanIf	CanIf/CanIfTrcvDrvCfg	
BSW Parameter		**BSW Type**
CanIfTrcvCfg		EcucParamConfContainerDef
BSW Description		

This container contains the configuration (parameters) of one addressed CAN transceiver by the underlying CAN Transceiver Driver module. For each CAN transceiver a seperate instance of this container has to be provided.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
CanIf	CanIf/CanIfTrcvDrvCfg/CanIfTrcvCfg	
BSW Parameter		**BSW Type**
CanIfTrcvCanTrcvRef		EcucSymbolicNameReferenceDef
BSW Description		
This parameter references to the logical handle of the underlying CAN transceiver from the CAN transceiver driver module to be served by the CAN Interface module. Range: 0..max. number of underlying supported CAN transceivers		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
CanIf	CanIf/CanIfTrcvDrvCfg/CanIfTrcvCfg	
BSW Parameter		**BSW Type**
CanIfTrcvId		EcucIntegerParamDef
BSW Description		
This parameter abstracts from the CAN Transceiver Driver specific parameter Transceiver. Each transceiver of all connected CAN Transceiver Driver modules shall be assigned to one specific TransceiverId of the CanIf. Range: 0..number of configured transceivers of all CAN Transceiver Driver modules		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
CanIf	CanIf/CanIfTrcvDrvCfg/CanIfTrcvCfg	
BSW Parameter		**BSW Type**
CanIfTrcvWakeupSupport		EcucBooleanParamDef
BSW Description		

This parameter defines if a respective transceiver of the referenced CAN Transceiver Driver modules is queriable for wake up events. True: Enabled False: Disabled	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

D.3 CanNm Mapping

BSW Module	**BSW Context**
CanNm	CanNm
BSW Parameter	**BSW Type**
CanNmGlobalConfig	EcucParamConfContainerDef
BSW Description	
This container contains the global configuration parameter of the CanNm. The parameters and the parameters of the sub containers shall be mapped to the C data type CanNm_ConfigType (for parameters where it is possible) which is passed to the CanNm_Init function. This container is a MultipleConfigurationContainer (only for variant 3), i.e. this container and its sub-containers exit once per configuration set.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**
CanNm	CanNm/CanNmGlobalConfig
BSW Parameter	**BSW Type**
CanNmBusLoadReductionEnabled	EcucBooleanParamDef
BSW Description	
Pre-processor switch for enabling busload reduction support.	
M2 Template	**M2 Description**
SystemTemplate	Enables busload reduction support
M2 Parameter	
NetworkManagement::NmClusterCoupling.nmBusloadReductionEnabled	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	**BSW Context**
CanNm	CanNm/CanNmGlobalConfig
BSW Parameter	**BSW Type**
CanNmBusSynchronizationEnabled	EcucBooleanParamDef
BSW Description	

Pre-processor switch for enabling bus synchronization support. This feature is required for gateway nodes only. calculationFormula = If (CanNmPassiveModeEnabled == False) then Equal(NmBusSynchronizationEnabled) else Equal(False)	
M2 Template	**M2 Description**
SystemTemplate	Enables bus synchronization support.
M2 Parameter	
NetworkManagement::NmEcu.nmBusSynchronizationEnabled	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	**BSW Context**
CanNm	CanNm/CanNmGlobalConfig
BSW Parameter	**BSW Type**
CanNmChannelConfig	EcucParamConfContainerDef
BSW Description	
This container contains the channel specific configuration parameter of the CanNm.	
M2 Template	**M2 Description**
SystemTemplate	Can specific NmCluster attributes
M2 Parameter	
NetworkManagement::CanNmCluster	
Mapping Rule	**Mapping Type**
Create container for each existing CanNmCluster.	full

BSW Module	**BSW Context**
CanNm	CanNm/CanNmGlobalConfig/CanNmChannelConfig
BSW Parameter	**BSW Type**
CanNmAllNmMessagesKeepAwake	EcucBooleanParamDef
BSW Description	
Specifies if CanNm drops irrelevant NM messages. false: Only NM messages with an with CRI bit = true and containing an PN request for this ECU triggers the standard RX indication handling true: Every NM message triggers the standard RX indication handling	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**
CanNm	CanNm/CanNmGlobalConfig/CanNmChannelConfig
BSW Parameter	**BSW Type**
CanNmBusLoadReductionActive	EcucBooleanParamDef
BSW Description	
This parameter defines if bus load reduction for the respective NM channel is active or not.	
M2 Template	**M2 Description**
SystemTemplate	It determines if bus load reduction for the respective CanNm channel is active or not.

M2 Parameter	
NetworkManagement::CanNmCluster.nmBusloadReductionActive	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	BSW Context	
CanNm	CanNm/CanNmGlobalConfig/CanNmChannelConfig	
BSW Parameter		**BSW Type**
CanNmCarWakeUpBitPosition		EcucIntegerParamDef
BSW Description		
Specifies the Bit position of the CWU within the NM-Message.		
M2 Template	**M2 Description**	
SystemTemplate	Specifies the bit position of the CarWakeUp within the NM-Message.	
M2 Parameter		
NetworkManagement::CanNmCluster.nmCarWakeUpBitPosition		
Mapping Rule		**Mapping Type**
The position of the Car Wakeup bit in the Ecuc is defined by the configuration parameters CanNmCarWakeUpBytePosition and CanNmCarWakeUpBitPosition (position in wakeUpByte). In the SysT the position is described only by the bit position in the NmMessage.		full

BSW Module	BSW Context	
CanNm	CanNm/CanNmGlobalConfig/CanNmChannelConfig	
BSW Parameter		**BSW Type**
CanNmCarWakeUpBytePosition		EcucIntegerParamDef
BSW Description		
Specifies the Byte position of the CWU within the NM-Message.		
M2 Template	**M2 Description**	
SystemTemplate	Specifies the bit position of the CarWakeUp within the NM-Message.	
M2 Parameter		
NetworkManagement::CanNmCluster.nmCarWakeUpBitPosition		
Mapping Rule		**Mapping Type**
The position of the Car Wakeup bit in the Ecuc is defined by the configuration parameters CanNmCarWakeUpBytePosition and CanNmCarWakeUpBitPosition (position in wakeUpByte). In the SysT the position is described only by the bit position in the NmMessage.		full

BSW Module	BSW Context	
CanNm	CanNm/CanNmGlobalConfig/CanNmChannelConfig	
BSW Parameter		**BSW Type**
CanNmCarWakeUpFilterEnabled		EcucBooleanParamDef
BSW Description		
If CWU filtering is supported, only the CWU bit within the NM message with source node identifier CanNmCarWakeUpFilterNodeId is considered as CWU request. FALSE - CWU filtering is not supported TRUE - CWU filtering is supported		
M2 Template	**M2 Description**	
SystemTemplate	If this attribute is set to true the CareWakeUp filtering is supported. In this case only the CarWakeUp bit within the NM message with source node identifier nmCarWakeUpFilterNodeId is considered as CarWakeUp request.	
M2 Parameter		

NetworkManagement::CanNmCluster.nmCarWakeUpFilterEnabled	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	**BSW Context**	
CanNm	CanNm/CanNmGlobalConfig/CanNmChannelConfig	
BSW Parameter		**BSW Type**
CanNmCarWakeUpFilterNodeId		EcucIntegerParamDef
BSW Description		
Source node identifier for CWU filtering. If CWU filtering is supported, only the CWU bit within the NM message with source node identifier CanNmCarWakeUpFilterNodeId is considered as CWU request.		
M2 Template	**M2 Description**	
SystemTemplate	Source node identifier for CarWakeUp filtering. If CarWakeUp filtering is supported (nmCarWakeUpFilterEnabled), only the CarWakeUp bit within the NM message with source node identifier nmCarWakeUpFilterNodeId is considered as CarWakeUp request.	
M2 Parameter		
NetworkManagement::CanNmCluster.nmCarWakeUpFilterNodeId		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
CanNm	CanNm/CanNmGlobalConfig/CanNmChannelConfig	
BSW Parameter		**BSW Type**
CanNmCarWakeUpRxEnabled		EcucBooleanParamDef
BSW Description		
Enables or disables support of CarWakeUp bit evaluation in received NM messages. FALSE - CarWakeUp not supported TRUE - CarWakeUp supported		
M2 Template	**M2 Description**	
SystemTemplate	If set to true this attribute enables the support of CarWakeUp bit evaluation in received NM messages.	
M2 Parameter		
NetworkManagement::CanNmCluster.nmCarWakeUpRxEnabled		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
CanNm	CanNm/CanNmGlobalConfig/CanNmChannelConfig	
BSW Parameter		**BSW Type**
CanNmComMNetworkHandleRef		EcucSymbolicNameReferenceDef
BSW Description		
This reference points to the unique channel defined by the ComMChannel and provides access to the unique channel index value in ComMChannelId.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
CanNm	CanNm/CanNmGlobalConfig/CanNmChannelConfig	
BSW Parameter		**BSW Type**
CanNmImmediateNmCycleTime		EcucFloatParamDef
BSW Description		
Defines the immediate NM PDU cycle time in seconds which is used for CanNmImmediateNmTransmissions NM PDU transmissions. This parameter is only valid if CanNmImmediateNmTransmissions is greater one.		
M2 Template	**M2 Description**	
SystemTemplate	Timeout of a NM message in seconds. It determines how long the NM shall wait with notification of transmission failure while communication errors occur on the bus.	
M2 Parameter		
NetworkManagement::CanNmCluster.nmImmediateNmCycleTime		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
CanNm	CanNm/CanNmGlobalConfig/CanNmChannelConfig	
BSW Parameter		**BSW Type**
CanNmImmediateNmTransmissions		EcucIntegerParamDef
BSW Description		
Defines the number of immediate NM PDUs which shall be transmitted. If the value is zero no immediate NM PDUs are transmitted. The cycle time of immeditate NM PDUs is defined by CanNmImmediateNmCycleTime.		
M2 Template	**M2 Description**	
SystemTemplate	Defines the number of immediate NM PDUs which shall be transmitted. If the value is zero no immediate NM PDUs are transmitted. The cycle time of immeditate NM PDUs is defined by nmImmediateNmCycleTime.	
M2 Parameter		
NetworkManagement::CanNmCluster.nmImmediateNmTransmissions		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
CanNm	CanNm/CanNmGlobalConfig/CanNmChannelConfig	
BSW Parameter		**BSW Type**
CanNmMsgCycleOffset		EcucFloatParamDef
BSW Description		
Time offset in the periodic transmission node. It determines the start delay of the transmission. Specified in seconds. This parameter is only valid if CanNmPassiveModeEnabled is False.		
M2 Template	**M2 Description**	
SystemTemplate	Node specific time offset in the periodic transmission node. It determines the start delay of the transmission. Specified in seconds.	
M2 Parameter		
NetworkManagement::CanNmNode.nmMsgCycleOffset		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context		
CanNm	CanNm/CanNmGlobalConfig/CanNmChannelConfig		
BSW Parameter		**BSW Type**	
CanNmMsgCycleTime		EcucFloatParamDef	
BSW Description			
Period of a NM-message in seconds. It determines the periodic rate in the "periodic transmission mode with bus load reduction" and is the basis for transmit scheduling in the "periodic transmission mode without bus load reduction". This parameter is only valid if CanNmPassiveModeEnabled is False.			
M2 Template	**M2 Description**		
SystemTemplate	Period of a CanNm message in seconds. It determines the periodic rate in the periodic transmission mode with bus load reduction and is the basis for transmit scheduling in the periodic transmission mode without bus load reduction.		
M2 Parameter			
NetworkManagement::CanNmCluster.nmMsgCycleTime			
Mapping Rule			**Mapping Type**
1:1 mapping			full

BSW Module	BSW Context		
CanNm	CanNm/CanNmGlobalConfig/CanNmChannelConfig		
BSW Parameter		**BSW Type**	
CanNmMsgReducedTime		EcucFloatParamDef	
BSW Description			
Node specific bus cycle time in the periodic transmission mode with bus load reduction. Specified in seconds. This parameter is only valid if CanNmBusLoadReductionEnabled == True and CanNmBusLoadReductionActive == True and CanNmPassiveModeEnabled == False Otherwise this parameter is not used.			
M2 Template	**M2 Description**		
SystemTemplate	Node specific bus cycle time in the periodic transmission mode with bus load reduction. Specified in seconds.		
M2 Parameter			
NetworkManagement::CanNmNode.nmMsgReducedTime			
Mapping Rule			**Mapping Type**
1:1 mapping			full

BSW Module	BSW Context	
CanNm	CanNm/CanNmGlobalConfig/CanNmChannelConfig	
BSW Parameter		**BSW Type**
CanNmMsgTimeoutTime		EcucFloatParamDef
BSW Description		
Transmission Timeout of NM-message. If there is no transmission confirmation by the CAN Interface within this timeout, the CANNM module shall give an error notification. This parameter is only valid if CANNM_PASSIVE_MODE_ENABLED is disabled.		
M2 Template	**M2 Description**	

— AUTOSAR CONFIDENTIAL —

SystemTemplate	Timeout of a NM message in seconds. It determines how long the NM shall wait with notification of transmission failure while communication errors occur on the bus.	
M2 Parameter		
NetworkManagement::CanNmCluster.nmMessageTimeoutTime		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
CanNm	CanNm/CanNmGlobalConfig/CanNmChannelConfig	
BSW Parameter		**BSW Type**
CanNmNodeId		EcucIntegerParamDef
BSW Description		
Node identifier of local node. This parameter is only valid if CanNmPassiveModeEnabled = False and CanNmNodeDetectionEnabled = True		
M2 Template	**M2 Description**	
SystemTemplate	Node identifier of local NmNode. Must be unique in the NmCluster.	
M2 Parameter		
NetworkManagement::CanNmNode.nmNodeId		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
CanNm	CanNm/CanNmGlobalConfig/CanNmChannelConfig	
BSW Parameter		**BSW Type**
CanNmPduCbvPosition		EcucEnumerationParamDef
BSW Description		

Defines the position of the control bit vector within the NM PDU.
 	
The value of the parameter represents the location of the control bit vector in the NM PDU (CanNmPduByte0 means byte 0, CanNmPduByte1 means byte 1, CanNmPduOff means source node identifier is not part of the NM PDU)
 	
if(CANNM_PDU_CBV_POSITION != CANNM_PDU_OFF && CANNM_PDU_NID_POSITION != CANNM_PDU_OFF) then CANNM_PDU_CBV_POSITION != CANNM_PDU_NID_POSITION
 	
if(CANNM_PDU_CBV_POSITION != CANNM_PDU_OFF && CANNM_PDU_NID_POSITION == CANNM_PDU_OFF) then CANNM_PDU_CBV_POSITION = CANNM_PDU_BYTE0
 	
ImplementationType: CanNm_PduPositionType		

M2 Template	**M2 Description**	
SystemTemplate	Defines the position of the control bit vector within the NM PDU (Bitpositon).	
M2 Parameter		
NetworkManagement::CanNmCluster.nmCbvPosition		
Mapping Rule		**Mapping Type**
Derive byte position from nmCbvPosition attribute. If this optional attribute is missing set CANNM_PDU_OFF as value.		full

BSW Module	BSW Context	
CanNm	CanNm/CanNmGlobalConfig/CanNmChannelConfig	
BSW Parameter		**BSW Type**
CanNmPduNidPosition		EcucEnumerationParamDef
BSW Description		

Defines the position of the source node identifier within the NM PDU.
 	
The value of the parameter represents the location of the source node identifier in the NM PDU (CanNMPduByte0 means byte 0, CanNmPduByte1 means byte 1, CanNmPduOff means source node identifier is not part of the NM PDU)
 	
if(CANNM_PDU_NID_POSITION != CANNM_PDU_OFF && CANNM_PDU_CBV_POSITION != CANNM_PDU_OFF) then CANNM_PDU_NID_POSITION != CANNM_PDU_CBV_POSITION
 	
if(CANNM_PDU_NID_POSITION != CANNM_PDU_OFF && CANNM_PDU_CBV_POSITION == CANNM_PDU_OFF) then CANNM_PDU_NID_POSITION = CANNM_PDU_BYTE0
 	
ImplementationType: CanNm_PduPositionType		

M2 Template	**M2 Description**	
SystemTemplate	Defines the bitposition of the source node identifier within the NM PDU.	
M2 Parameter		
NetworkManagement::CanNmCluster.nmNidPosition		
Mapping Rule		**Mapping Type**
Derive byte position from nmNidPosition attribute. If this optional attribute is missing set CANNM_PDU_OFF as value.		full

BSW Module	BSW Context	
CanNm	CanNm/CanNmGlobalConfig/CanNmChannelConfig	
BSW Parameter		**BSW Type**
CanNmPnEnabled		EcucBooleanParamDef
BSW Description		
Enables or disables support of partial networking. false: Partial networking Range not supported true: Partial networking supported		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
CanNm	CanNm/CanNmGlobalConfig/CanNmChannelConfig	
BSW Parameter		**BSW Type**
CanNmPnEraCalcEnabled		EcucBooleanParamDef
BSW Description		
Specifies if CanNm calculates the PN request information for external requests. (ERA) false: PN request are not calculated true: PN request are calculated		
M2 Template	**M2 Description**	

M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**	
CanNm	CanNm/CanNmGlobalConfig/CanNmChannelConfig	
BSW Parameter		**BSW Type**
CanNmPnEraRxNSduRef		EcucReferenceDef
BSW Description		
Reference to a Pdu in the COM-Stack. The SduRef is required for every CanNm Channel, because ERA is reported per channel.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
CanNm	CanNm/CanNmGlobalConfig/CanNmChannelConfig	
BSW Parameter		**BSW Type**
CanNmPnHandleMultipleNetworkRequests		EcucBooleanParamDef
BSW Description		
false: CanNm_NetworkRequest is ignored in NO. true: CanNm_NetworkRequest triggers a change from NO to RM.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
CanNm	CanNm/CanNmGlobalConfig/CanNmChannelConfig	
BSW Parameter		**BSW Type**
CanNmRemoteSleepIndTime		EcucFloatParamDef
BSW Description		
Timeout for Remote Sleep Indication. It defines the time in seconds how long it shall take to recognize that all other nodes are ready to sleep. Typically it should be equal to: n * CanNmMsgCycleTime, where n denotes the number of NM-Messages that are normally sent before Remote Sleep Indication is detected. The value of n decremented by one determines the amount of lost NM-Messages that can be tolerated by the Remote Sleep Indication procedure. The value 0 denotes that no Remote Sleep Indication functionality is configured.		
M2 Template	**M2 Description**	
SystemTemplate	Timeout for Remote Sleep Indication in seconds. It defines the time how long it shall take to recognize that all other nodes are ready to sleep.	

M2 Parameter	
NetworkManagement::CanNmCluster.nmRemoteSleepIndTime	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	BSW Context	
CanNm	CanNm/CanNmGlobalConfig/CanNmChannelConfig	
BSW Parameter		**BSW Type**
CanNmRepeatMessageTime		EcucFloatParamDef
BSW Description		
Timeout for Repeat Message State. It defines the time in seconds how long the NM shall stay in the Repeat Message State. Typically it should be equal to: n * CanNmMsgCycleTime, where n denotes the number of NM-Messages that are normally sent in the Repeat Message State. The value of n decremented by one determines the amount of lost NM-Messages that can be tolerated by the node detection procedure. The value 0 denotes that no Repeat Message State is configured. It means that Repeat Message State is transient what implicates that it is left immediately after entrance and in result no start-up stability is guaranteed and no node detection procedure is possible.		
M2 Template	**M2 Description**	
SystemTemplate	Timeout for Repeat Message State in seconds. Defines the time how long the NM shall stay in the Repeat Message State.	
M2 Parameter		
NetworkManagement::CanNmCluster.nmRepeatMessageTime		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
CanNm	CanNm/CanNmGlobalConfig/CanNmChannelConfig	
BSW Parameter		**BSW Type**
CanNmRxPdu		EcucParamConfContainerDef
BSW Description		
This container is used to configure the Rx PDU properties that are used for the CanNm Channel.		
M2 Template	**M2 Description**	
SystemTemplate	Receive NM Pdu	
M2 Parameter		
NetworkManagement::CanNmNode.rxNmPdu		
Mapping Rule		**Mapping Type**
Create container for each NmPdu that is received on the regarded Nm cluster		full

BSW Module	BSW Context	
CanNm	CanNm/CanNmGlobalConfig/CanNmChannelConfig/CanNmRxPdu	
BSW Parameter		**BSW Type**
CanNmRxPduId		EcucIntegerParamDef
BSW Description		
This parameter defines the Rx PDU ID of the CanIf L-PDU range that is associated with this CanNmChannel instance.		
M2 Template	**M2 Description**	
M2 Parameter		

Mapping Rule	Mapping Type
	local

BSW Module	BSW Context
CanNm	CanNm/CanNmGlobalConfig/CanNmChannelConfig/CanNmRxPdu
BSW Parameter	**BSW Type**
CanNmRxPduRef	EcucReferenceDef
BSW Description	
Reference to the global PDU that is used by this CanNmChannel instance.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	BSW Context
CanNm	CanNm/CanNmGlobalConfig/CanNmChannelConfig
BSW Parameter	**BSW Type**
CanNmTimeoutTime	EcucFloatParamDef
BSW Description	
Network Timeout for NM-Messages. It denotes the time in seconds how long the NM shall stay in the Network Mode before transition into Prepare Bus-Sleep Mode shall take place. It shall be equal for all nodes in the cluster. It shall be greater than CanNmMsgCycleTime. Typically it should be equal to: n * CanNmMsgCycleTime, where n denotes the number of NM-Message cycle times in the Ready Sleep State before transition into the Bus-Sleep Mode is initiated. The value of n decremented by one determines the amount of lost NM-Messages that can be tolerated by the coordination algorithm.	
M2 Template	**M2 Description**
SystemTemplate	Network Timeout for CanNm PDUs in seconds. It denotes the time how long the CanNm shall stay in the Network Mode before transition into Prepare Bus-Sleep Mode shall take place.
M2 Parameter	
NetworkManagement::CanNmCluster.nmNetworkTimeout	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	BSW Context
CanNm	CanNm/CanNmGlobalConfig/CanNmChannelConfig
BSW Parameter	**BSW Type**
CanNmTxPdu	EcucParamConfContainerDef
BSW Description	
This container contains the CanNmTxConfirmationPduId and the CanNmTxPduRef.	
M2 Template	**M2 Description**
SystemTemplate	Transmit NM Pdu
M2 Parameter	
NetworkManagement::CanNmNode.txNmPdu	

Mapping Rule	Mapping Type
Create container for each NmPdu that is transmitted on the regarded Nmcluster	full

BSW Module	BSW Context	
CanNm	CanNm/CanNmGlobalConfig/CanNmChannelConfig/CanNmTxPdu	
BSW Parameter		**BSW Type**
CanNmTxConfirmationPduId		EcucIntegerParamDef
BSW Description		
Handle Id to be used by the Lower Layer to confirm the transmission of the CanNmTxPdu to the LowerLayer.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
CanNm	CanNm/CanNmGlobalConfig/CanNmChannelConfig/CanNmTxPdu	
BSW Parameter		**BSW Type**
CanNmTxPduRef		EcucReferenceDef
BSW Description		
The reference to the common PDU structure.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
CanNm	CanNm/CanNmGlobalConfig/CanNmChannelConfig	
BSW Parameter		**BSW Type**
CanNmUserDataLength		EcucIntegerParamDef
BSW Description		
Defines the length of the user data contained in the NM PDU		
M2 Template	**M2 Description**	
SystemTemplate	Defines the length of the user data contained in the NM Pdu.	
M2 Parameter		
NetworkManagement::CanNmCluster.nmUserDataLength		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
CanNm	CanNm/CanNmGlobalConfig/CanNmChannelConfig	
BSW Parameter		**BSW Type**
CanNmUserDataTxPdu		EcucParamConfContainerDef
BSW Description		

This optional container is used to configure the UserNm PDU. This container is only available if CanNmComUserDataSupport is enabled.		
M2 Template	**M2 Description**	
SystemTemplate	This optional aggregation is used to describe NmUserData that is transmitted in the NmPdu.	
M2 Parameter		
NetworkManagement::CanNmPdu.iSignalToIPduMapping		
Mapping Rule		**Mapping Type**
Create container for each NmPdu that aggregates the ISignalToIPduMapping element. The configuration for these Pdus (e.g. Transfer Properties) shall be derived from this information.		full

BSW Module	**BSW Context**	
CanNm	CanNm/CanNmGlobalConfig/CanNmChannelConfig/CanNmUserDataTxPdu	
BSW Parameter		**BSW Type**
CanNmTxUserDataPduId		EcucIntegerParamDef
BSW Description		
This parameter defines the Handle ID of the NM User Data I-PDU.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
CanNm	CanNm/CanNmGlobalConfig/CanNmChannelConfig/CanNmUserDataTxPdu	
BSW Parameter		**BSW Type**
CanNmTxUserDataPduRef		EcucReferenceDef
BSW Description		
Reference to the NM User Data I-PDU in the global PDU collection.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
CanNm	CanNm/CanNmGlobalConfig/CanNmChannelConfig	
BSW Parameter		**BSW Type**
CanNmWaitBusSleepTime		EcucFloatParamDef
BSW Description		
Timeout for bus calm down phase. It denotes the time in seconds how long the NM shall stay in the Prepare Bus-Sleep Mode before transition into Bus-Sleep Mode shall take place. It shall be equal for all nodes in the cluster. It shall be long enough to make all Tx-buffer empty.		
M2 Template	**M2 Description**	

SystemTemplate	Timeout for bus calm down phase in seconds. It denotes the time how long the CanNm shall stay in the Prepare Bus-Sleep Mode before transition into Bus-Sleep Mode shall take place.
M2 Parameter	
NetworkManagement::CanNmCluster.nmWaitBusSleepTime	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	**BSW Context**
CanNm	CanNm/CanNmGlobalConfig
BSW Parameter	**BSW Type**
CanNmComControlEnabled	EcucBooleanParamDef
BSW Description	
Pre-processor switch for enabling the Communication Control support. calculationformula = Equal(NmComControlEnabled)	
M2 Template	**M2 Description**
SystemTemplate	Enables the Communication Control support.
M2 Parameter	
NetworkManagement::NmEcu.nmComControlEnabled	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	**BSW Context**
CanNm	CanNm/CanNmGlobalConfig
BSW Parameter	**BSW Type**
CanNmComUserDataSupport	EcucBooleanParamDef
BSW Description	
Enable/disable the user data support.	
M2 Template	**M2 Description**
SystemTemplate	Enable/disable the user data support.
M2 Parameter	
FibexCore::CoreCommunication::NmPdu.nmDataInformation	
Mapping Rule	**Mapping Type**
If the nmDataInformation attribute is set to true for NmPdus that are transmitted by the regarded Ecu than this parameter must also be set to true.	full

BSW Module	**BSW Context**
CanNm	CanNm/CanNmGlobalConfig
BSW Parameter	**BSW Type**
CanNmCoordinatorSyncSupport	EcucBooleanParamDef
BSW Description	
Enables/disables the coordinator synchronisation support.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	BSW Context	
CanNm	CanNm/CanNmGlobalConfig	
BSW Parameter		**BSW Type**
CanNmDevErrorDetect		EcucBooleanParamDef
BSW Description		
Pre-processor switch for enabling development error detection support.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
CanNm	CanNm/CanNmGlobalConfig	
BSW Parameter		**BSW Type**
CanNmImmediateRestartEnabled		EcucBooleanParamDef
BSW Description		
Pre-processor switch for enabling the asynchronous transmission of a NM PDU upon bus-communication request in Prepare-Bus-Sleep mode.		
M2 Template	**M2 Description**	
SystemTemplate	Enables the asynchronous transmission of a CanNm PDU upon bus-communication request in Prepare-Bus-Sleep mode.	
M2 Parameter		
NetworkManagement::NmClusterCoupling.nmImmediateRestartEnabled		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
CanNm	CanNm/CanNmGlobalConfig	
BSW Parameter		**BSW Type**
CanNmImmediateTxconfEnabled		EcucBooleanParamDef
BSW Description		
Enable/disable the immediate tx confirmation.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
CanNm	CanNm/CanNmGlobalConfig	
BSW Parameter		**BSW Type**
CanNmMainFunctionPeriod		EcucFloatParamDef
BSW Description		
Call cycle in seconds of CanNm_MainFunction.		
M2 Template	**M2 Description**	
M2 Parameter		

Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**	
CanNm	CanNm/CanNmGlobalConfig	
BSW Parameter		**BSW Type**
CanNmNodeDetectionEnabled		EcucBooleanParamDef
BSW Description		
Precompile time switch to enable the node detection feature.		
M2 Template	**M2 Description**	
SystemTemplate	Enables the Request Repeat Message Request support. Only valid if nmNodeIdEnabled is set to true.	
M2 Parameter		
NetworkManagement::NmEcu.nmNodeDetectionEnabled		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
CanNm	CanNm/CanNmGlobalConfig	
BSW Parameter		**BSW Type**
CanNmNumberOfChannels		EcucIntegerParamDef
BSW Description		
Number of Can NM channels allowed within one ECU.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
CanNm	CanNm/CanNmGlobalConfig	
BSW Parameter		**BSW Type**
CanNmPassiveModeEnabled		EcucBooleanParamDef
BSW Description		
Pre-processor switch for enabling support of the Passive Mode. calculationFormula = Equal(NmPassiveModeEnabled)		
M2 Template	**M2 Description**	
SystemTemplate	Enables support of the Passive Mode.	
M2 Parameter		
NetworkManagement::NmEcu.nmPassiveModeEnabled		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
CanNm	CanNm/CanNmGlobalConfig	
BSW Parameter		**BSW Type**

CanNmPduRxIndicationEnabled		EcucBooleanParamDef	
BSW Description			
Pre-processor switch for enabling the PDU Rx Indication. calculationFormula = Equal(NmPduRxIndicationEnabled)			
M2 Template	**M2 Description**		
SystemTemplate	Switch for enabling the PDU Rx Indication.		
M2 Parameter			
NetworkManagement::NmEcu.nmPduRxIndicationEnabled			
Mapping Rule			**Mapping Type**
1:1 mapping			full

BSW Module	**BSW Context**		
CanNm	CanNm/CanNmGlobalConfig		
BSW Parameter		**BSW Type**	
CanNmPnEiraCalcEnabled		EcucBooleanParamDef	
BSW Description			
Specifies if CanNm calculates the PN request information for internal an external requests. (EIRA) true: PN request are calculated false: PN request are not calculated			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	**BSW Context**		
CanNm	CanNm/CanNmGlobalConfig		
BSW Parameter		**BSW Type**	
CanNmPnEiraRxNSduRef		EcucReferenceDef	
BSW Description			
Reference to a Pdu in the COM-Stack. Only one SduRef is required for CanNm because the EIRA is the aggregation over all Can Channels.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	**BSW Context**	
CanNm	CanNm/CanNmGlobalConfig	
BSW Parameter		**BSW Type**
CanNmPnInfo		EcucParamConfContainerDef
BSW Description		
PN information configuration		
M2 Template	**M2 Description**	
M2 Parameter		

Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**	
CanNm	CanNm/CanNmGlobalConfig/CanNmPnInfo	
BSW Parameter		**BSW Type**
CanNmPnFilterMaskByte		EcucParamConfContainerDef
BSW Description		
PN information configuration		
M2 Template	**M2 Description**	
SystemTemplate		
M2 Parameter		
CanTopology::CanCommunicationConnector.pncWakeupDataMask		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
CanNm	CanNm/CanNmGlobalConfig/CanNmPnInfo/CanNmPnFilterMaskByte	
BSW Parameter		**BSW Type**
CanNmPnFilterMaskByteIndex		EcucIntegerParamDef
BSW Description		
Specifies the offset of the PN request information in the NM message.		
M2 Template	**M2 Description**	
SystemTemplate		
M2 Parameter		
CanTopology::CanCommunicationConnector.pncWakeupDataMask		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
CanNm	CanNm/CanNmGlobalConfig/CanNmPnInfo/CanNmPnFilterMaskByte	
BSW Parameter		**BSW Type**
CanNmPnFilterMaskByteValue		EcucIntegerParamDef
BSW Description		
Parameter to configure the filter mask byte.		
M2 Template	**M2 Description**	
SystemTemplate		
M2 Parameter		
CanTopology::CanCommunicationConnector.pncWakeupDataMask		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
CanNm	CanNm/CanNmGlobalConfig/CanNmPnInfo	
BSW Parameter		**BSW Type**
CanNmPnInfoLength		EcucIntegerParamDef
BSW Description		
Specifies the length of the PN request information in the NM message.		

— AUTOSAR CONFIDENTIAL —

M2 Template	M2 Description	
SystemTemplate	Length of the partial networking request release information vector.	
M2 Parameter		
System.pncVectorLength		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context		
CanNm	CanNm/CanNmGlobalConfig/CanNmPnInfo		
BSW Parameter		**BSW Type**	
CanNmPnInfoOffset		EcucIntegerParamDef	
BSW Description			
Specifies the offset of the PN request information in the NM message.			
M2 Template	**M2 Description**		
SystemTemplate	Absolute offset (with respect to the Frame) of the partial networking request release information vector.		
M2 Parameter			
System.pncVectorOffset			
Mapping Rule			**Mapping Type**
1:1 mapping			full

BSW Module	BSW Context		
CanNm	CanNm/CanNmGlobalConfig		
BSW Parameter		**BSW Type**	
CanNmPnResetTime		EcucFloatParamDef	
BSW Description			
Specifies the runtime of the reset timer in seconds. This reset time is valid for the reset of PN requests in the EIRA and in the ERA. The value shall be the same for every channel. Thus it is a global config parameter.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	BSW Context	
CanNm	CanNm/CanNmGlobalConfig	
BSW Parameter		**BSW Type**
CanNmRemoteSleepIndEnabled		EcucBooleanParamDef
BSW Description		
Pre-processor switch for enabling remote sleep indication support. This feature is required for gateway nodes only. calculationFormula = If (CanNmPassiveModeEnabled == False) then Equal(NmRemoteSleepIndEnabled) else Equal(False)		
M2 Template	**M2 Description**	
SystemTemplate	Switch for enabling remote sleep indication support.	
M2 Parameter		
NetworkManagement::NmEcu.nmRemoteSleepIndEnabled		

Mapping Rule	Mapping Type
1:1 mapping	full

BSW Module	BSW Context	
CanNm	CanNm/CanNmGlobalConfig	
BSW Parameter		**BSW Type**
CanNmRepeatMsgIndEnabled		EcucBooleanParamDef
BSW Description		
Enable/disable the notification that a RepeatMessageRequest bit has been received. calculationformula = Equal(NmRepeatMsgIndEnabled)		
M2 Template	**M2 Description**	
SystemTemplate	Enable/disable the notification that a RepeatMessageRequest bit has been received.	
M2 Parameter		
NetworkManagement::CanNmEcu.nmRepeatMsgIndicationEnabled		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
CanNm	CanNm/CanNmGlobalConfig	
BSW Parameter		**BSW Type**
CanNmStateChangeIndEnabled		EcucBooleanParamDef
BSW Description		
Pre-processor switch for enabling the CAN NM state change notification. calculationFormula = Equal(NmStateChangeIdEnabled)		
M2 Template	**M2 Description**	
SystemTemplate	Enables the CAN Network Management state change notification.	
M2 Parameter		
NetworkManagement::NmEcu.nmStateChangeIndEnabled		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
CanNm	CanNm/CanNmGlobalConfig	
BSW Parameter		**BSW Type**
CanNmUserDataEnabled		EcucBooleanParamDef
BSW Description		
Pre-processor switch for enabling user data support. calculationFormula = Equal(NmUserDataEnabled)		
M2 Template	**M2 Description**	
SystemTemplate	Switch for enabling user data support.	
M2 Parameter		
NetworkManagement::NmEcu.nmUserDataEnabled		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
CanNm	CanNm/CanNmGlobalConfig	
BSW Parameter		**BSW Type**
CanNmVersionInfoApi		EcucBooleanParamDef
BSW Description		
Pre-processor switch for enabling version info API support.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

D.4 CanTp Mapping

BSW Module	BSW Context	
CanTp	CanTp	
BSW Parameter		**BSW Type**
CanTpConfig		EcucParamConfContainerDef
BSW Description		
This container contains the configuration parameters and sub containers of the AUTOSAR CanTp module. This container is a MultipleConfigurationContainer, i.e. this container and its sub-containers exist once per configuration set.		
M2 Template	**M2 Description**	
SystemTemplate	This element defines exactly one CAN TP Configuration.	
M2 Parameter		
TransportProtocols::CanTpConfig		
Mapping Rule		**Mapping Type**
Create Container if CanTpConfig exists in ECU Extract.		full

BSW Module	BSW Context	
CanTp	CanTp/CanTpConfig	
BSW Parameter		**BSW Type**
CanTpChannel		EcucParamConfContainerDef
BSW Description		
This container contains the configuration parameters of the CanTp channel.		
M2 Template	**M2 Description**	
SystemTemplate	Configuration parameters of the CanTp channel.	
M2 Parameter		
TransportProtocols::CanTpChannel		
Mapping Rule		**Mapping Type**
Create Container ifor each CanTpChannel that exist in ECU Extract.		full

BSW Module	BSW Context	
CanTp	CanTp/CanTpConfig/CanTpChannel	
BSW Parameter		**BSW Type**
CanTpChannelMode		EcucEnumerationParamDef
BSW Description		
The CAN Transport Layer supports half and full duplex channel modes.		
M2 Template	**M2 Description**	

<table>
<tr><td>SystemTemplate</td><td colspan="2">The CAN Transport Layer supports half and full duplex channel modes.</td></tr>
<tr><td colspan="3">M2 Parameter</td></tr>
<tr><td colspan="3">TransportProtocols::CanTpChannel.channelMode</td></tr>
<tr><td colspan="2">Mapping Rule</td><td>Mapping Type</td></tr>
<tr><td colspan="2">1:1 mapping</td><td>full</td></tr>
</table>

<table>
<tr><td>BSW Module</td><td colspan="3">BSW Context</td></tr>
<tr><td>CanTp</td><td colspan="3">CanTp/CanTpConfig/CanTpChannel</td></tr>
<tr><td colspan="2">BSW Parameter</td><td colspan="2">BSW Type</td></tr>
<tr><td colspan="2">CanTpRxNSdu</td><td colspan="2">EcucParamConfContainerDef</td></tr>
<tr><td colspan="4">BSW Description</td></tr>
<tr><td colspan="4">The following parameters needs to be configured for each CAN N-SDU that the CanTp module receives via the CanTpChannel.</td></tr>
<tr><td>M2 Template</td><td colspan="3">M2 Description</td></tr>
<tr><td>SystemTemplate</td><td colspan="3">Reference to the IPdu that is segmented by the Transport Protocol.</td></tr>
<tr><td colspan="4">M2 Parameter</td></tr>
<tr><td colspan="4">TransportProtocols::CanTpConnection.tpSdu</td></tr>
<tr><td colspan="3">Mapping Rule</td><td>Mapping Type</td></tr>
<tr><td colspan="3">Create container for each existing CanTpConnection that contains a reference to an N-SDU that is received.</td><td>full</td></tr>
</table>

<table>
<tr><td>BSW Module</td><td colspan="3">BSW Context</td></tr>
<tr><td>CanTp</td><td colspan="3">CanTp/CanTpConfig/CanTpChannel/CanTpRxNSdu</td></tr>
<tr><td colspan="2">BSW Parameter</td><td colspan="2">BSW Type</td></tr>
<tr><td colspan="2">CanTpBs</td><td colspan="2">EcucIntegerParamDef</td></tr>
<tr><td colspan="4">BSW Description</td></tr>
<tr><td colspan="4">Sets the number of N-PDUs the CanTp receiver allows the sender to send, before waiting for an authorization to continue transmission of the following N-PDUs.For further details on this parameter value see ISO 15765-2 specification.</td></tr>
<tr><td>M2 Template</td><td colspan="3">M2 Description</td></tr>
<tr><td>SystemTemplate</td><td colspan="3">The maximum number of N-PDUs the CanTp receiver allows the sender to send, before waiting for an authorization to continue transmission of the following N-PDUs.</td></tr>
<tr><td colspan="4">M2 Parameter</td></tr>
<tr><td colspan="4">TransportProtocols::CanTpConnection.maxBlockSize</td></tr>
<tr><td colspan="3">Mapping Rule</td><td>Mapping Type</td></tr>
<tr><td colspan="3">1:1 mapping</td><td>full</td></tr>
</table>

<table>
<tr><td>BSW Module</td><td colspan="3">BSW Context</td></tr>
<tr><td>CanTp</td><td colspan="3">CanTp/CanTpConfig/CanTpChannel/CanTpRxNSdu</td></tr>
<tr><td colspan="2">BSW Parameter</td><td colspan="2">BSW Type</td></tr>
<tr><td colspan="2">CanTpNAe</td><td colspan="2">EcucParamConfContainerDef</td></tr>
<tr><td colspan="4">BSW Description</td></tr>
<tr><td colspan="4">Contains the parameters needed to configure each RxNSdu or TxNSdu with CanTpAddressingFormat set to CanTpMixed.</td></tr>
<tr><td>M2 Template</td><td colspan="3">M2 Description</td></tr>
<tr><td>SystemTemplate</td><td colspan="3">Declares which communication addressing mode is supported.</td></tr>
<tr><td colspan="4">M2 Parameter</td></tr>
<tr><td colspan="4">TransportProtocols::CanTpConnection.addressingFormat</td></tr>
<tr><td colspan="3">Mapping Rule</td><td>Mapping Type</td></tr>
<tr><td colspan="3">Create container if addressingFormat is set to "mixed".</td><td>full</td></tr>
</table>

<table>
<tr><td>BSW Module</td><td colspan="3">BSW Context</td></tr>
<tr><td>CanTp</td><td colspan="3">CanTp/CanTpConfig/CanTpChannel/CanTpRxNSdu/CanTpNAe</td></tr>
<tr><td colspan="2">BSW Parameter</td><td colspan="2">BSW Type</td></tr>
<tr><td colspan="2">CanTpNAe</td><td colspan="2">EcucIntegerParamDef</td></tr>
<tr><td colspan="4">BSW Description</td></tr>
<tr><td colspan="4">If an RxNsdu or a TxNsdu is configured for mixed addressing
format, this parameter contains the transport protocol address extension value.</td></tr>
<tr><td>M2 Template</td><td colspan="3">M2 Description</td></tr>
<tr><td>SystemTemplate</td><td colspan="3">An ECUs TP address on the referenced channel. This represents the diagnostic Address.</td></tr>
<tr><td colspan="4">M2 Parameter</td></tr>
<tr><td colspan="4">TransportProtocols::TPAddress.tpAddressExtensionValue</td></tr>
<tr><td colspan="3">Mapping Rule</td><td>Mapping Type</td></tr>
<tr><td colspan="3">The CanTPConnection contains a reference to the SDU and a relation to the Tp Node that contains the TpAddressExtension.</td><td>full</td></tr>
</table>

<table>
<tr><td>BSW Module</td><td colspan="3">BSW Context</td></tr>
<tr><td>CanTp</td><td colspan="3">CanTp/CanTpConfig/CanTpChannel/CanTpRxNSdu</td></tr>
<tr><td colspan="2">BSW Parameter</td><td colspan="2">BSW Type</td></tr>
<tr><td colspan="2">CanTpNSa</td><td colspan="2">EcucParamConfContainerDef</td></tr>
<tr><td colspan="4">BSW Description</td></tr>
<tr><td colspan="4">Contains the parameters needed to configure each RxNSdu or TxNSdu with CanTpAddressingFormat set to CanTpExtended.</td></tr>
<tr><td>M2 Template</td><td colspan="3">M2 Description</td></tr>
<tr><td>SystemTemplate</td><td colspan="3">Declares which communication addressing mode is supported.</td></tr>
<tr><td colspan="4">M2 Parameter</td></tr>
<tr><td colspan="4">TransportProtocols::CanTpConnection.addressingFormat</td></tr>
<tr><td colspan="3">Mapping Rule</td><td>Mapping Type</td></tr>
<tr><td colspan="3">Create container if addressingFormat is set to "extended".</td><td>full</td></tr>
</table>

<table>
<tr><td>BSW Module</td><td colspan="3">BSW Context</td></tr>
<tr><td>CanTp</td><td colspan="3">CanTp/CanTpConfig/CanTpChannel/CanTpRxNSdu/CanTpNSa</td></tr>
<tr><td colspan="2">BSW Parameter</td><td colspan="2">BSW Type</td></tr>
<tr><td colspan="2">CanTpNSa</td><td colspan="2">EcucIntegerParamDef</td></tr>
<tr><td colspan="4">BSW Description</td></tr>
<tr><td colspan="4">If an RxNSdu or a TxNSdu is configured for extended addressing format, this parameter contains the transport protocol source address's value.</td></tr>
<tr><td>M2 Template</td><td colspan="3">M2 Description</td></tr>
<tr><td>System Template</td><td colspan="3">An ECUs TP address on the referenced channel. This represents the diagnostic Address.</td></tr>
<tr><td colspan="4">M2 Parameter</td></tr>
<tr><td colspan="4">TransportProtocols::TPAddress.tpAddress</td></tr>
<tr><td colspan="3">Mapping Rule</td><td>Mapping Type</td></tr>
<tr><td colspan="3">The CanTPConnection contains a reference to the SDU and a relation to the Tp Node that contains the TpAddress.</td><td>full</td></tr>
</table>

<table>
<tr><td>BSW Module</td><td colspan="3">BSW Context</td></tr>
<tr><td>CanTp</td><td colspan="3">CanTp/CanTpConfig/CanTpChannel/CanTpRxNSdu</td></tr>
<tr><td colspan="2">BSW Parameter</td><td colspan="2">BSW Type</td></tr>
<tr><td colspan="2">CanTpNTa</td><td colspan="2">EcucParamConfContainerDef</td></tr>
</table>

— AUTOSAR CONFIDENTIAL —

BSW Description	
The following parameters need to be configured for each RxNsdu or TxNsdu with the CanTpAddressingFormat set to CanTpExtended.	
M2 Template	**M2 Description**
SystemTemplate	Declares which communication addressing mode is supported.
M2 Parameter	
TransportProtocols::CanTpConnection.addressingFormat	
Mapping Rule	**Mapping Type**
Create container if addressingFormat is set to "extended".	full

BSW Module	BSW Context
CanTp	CanTp/CanTpConfig/CanTpChannel/CanTpRxNSdu/CanTpNTa
BSW Parameter	**BSW Type**
CanTpNTa	EcucIntegerParamDef
BSW Description	
If an RxNsdu or a TxNsdu is configured for extended addressing format, this parameter contains the transport protocol target address's value.	
M2 Template	**M2 Description**
SystemTemplate	An ECUs TP address on the referenced channel. This represents the diagnostic Address.
M2 Parameter	
TransportProtocols::TPAddress.tpAddress	
Mapping Rule	**Mapping Type**
The CanTPConnection contains a reference to the SDU and a relation to the Tp Node that contains the TpAddress.	full

BSW Module	BSW Context
CanTp	CanTp/CanTpConfig/CanTpChannel/CanTpRxNSdu
BSW Parameter	**BSW Type**
CanTpNar	EcucFloatParamDef
BSW Description	
Value in seconds of the N_Ar timeout. N_Ar is the time for transmission of a CAN frame (any N_PDU) on the receiver side.	
M2 Template	**M2 Description**
SystemTemplate	This attribute states the timeout between the PDU transmit request of the Transport Layer to the FlexRay Interface and the corresponding confirmation of the FlexRay Interface on the receiver side (for FC or AF). Specified in seconds.
M2 Parameter	
TransportProtocols::CanTpNode.timeoutAr	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	BSW Context
CanTp	CanTp/CanTpConfig/CanTpChannel/CanTpRxNSdu
BSW Parameter	**BSW Type**
CanTpNbr	EcucFloatParamDef
BSW Description	
Value in seconds of the performance requirement for (N_Br + N_Ar). N_Br is the elapsed time between the receiving indication of a FF or CF or the transmit confirmation of a FC, until the transmit request of the next FC.	
M2 Template	**M2 Description**

SystemTemplate	Value in seconds of the performance requirement for (N_Br + N_Ar).
M2 Parameter	
TransportProtocols::CanTpConnection.timeoutBr	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	**BSW Context**
CanTp	CanTp/CanTpConfig/CanTpChannel/CanTpRxNSdu
BSW Parameter	**BSW Type**
CanTpNcr	EcucFloatParamDef
BSW Description	
Value in seconds of the N_Cr timeout. N_Cr is the time until reception of the next Consecutive Frame N_PDU.	
M2 Template	**M2 Description**
SystemTemplate	This parameter defines the timeout value for waiting for a CF or FF-x (in case of retry) after receiving the last CF or after sending an FC or AF on the receiver side. Specified in seconds.
M2 Parameter	
TransportProtocols::CanTpConnection.timeoutCr	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	**BSW Context**
CanTp	CanTp/CanTpConfig/CanTpChannel/CanTpRxNSdu
BSW Parameter	**BSW Type**
CanTpRxAddressingFormat	EcucEnumerationParamDef
BSW Description	
Declares which communication addressing mode is supported for this Rx N-SDU. Enum values: CanTpStandard. To use normal addressing format. CanTpExtended. To use extended addressing format. CanTpMixed. To use mixed addressing format.	
M2 Template	**M2 Description**
SystemTemplate	Declares which communication addressing mode is supported
M2 Parameter	
TransportProtocols::CanTpConnection.addressingFormat	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	**BSW Context**
CanTp	CanTp/CanTpConfig/CanTpChannel/CanTpRxNSdu
BSW Parameter	**BSW Type**
CanTpRxDl	EcucIntegerParamDef
BSW Description	
Data Length Code of this RxNsdu. In case of variable message length, this value indicates the minimum data length. Depending on SF or FF N-SDU the value will be limited to 7 (6 for an extended addressing format) and 4095 respectively.	
M2 Template	**M2 Description**

SystemTemplate	Maximum Pdu length in bits. In case of dynamic length IPdus (containing a dynamical length signal), this value indicates the minimum data length.	
M2 Parameter		
Pdu.length		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
CanTp	CanTp/CanTpConfig/CanTpChannel/CanTpRxNSdu	
BSW Parameter		**BSW Type**
CanTpRxNPdu		EcucParamConfContainerDef
BSW Description		
Used for grouping of the ID of a PDU and the Reference to a PDU.		
M2 Template	**M2 Description**	
SystemTemplate	Reference to an Data NPdu.	
M2 Parameter		
TransportProtocols::CanTpConnection.dataPdu		
Mapping Rule		**Mapping Type**
Create container if the CanTpConnection contains a reference to a DataNpdu that is received by the regarded ECU.		full

BSW Module	**BSW Context**	
CanTp	CanTp/CanTpConfig/CanTpChannel/CanTpRxNSdu/CanTpRxNPdu	
BSW Parameter		**BSW Type**
CanTpRxNPduId		EcucIntegerParamDef
BSW Description		
The N-PDU identifier attached to the RxNsdu is identified by CanTpRxNSduId. Each RxNsdu identifier is linked to only one SF/FF/CF N-PDU identifier. Nevertheless, in the case of extended or mixed addressing format, the same N-PDU identifier can be used for several N-SDU identifiers. The distinction is made by the N_TA or N_AE value (first data byte of SF or FF frames).		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
CanTp	CanTp/CanTpConfig/CanTpChannel/CanTpRxNSdu/CanTpRxNPdu	
BSW Parameter		**BSW Type**
CanTpRxNPduRef		EcucReferenceDef
BSW Description		
Reference to a Pdu in the COM-Stack.		
M2 Template	**M2 Description**	
M2 Parameter		

Mapping Rule	Mapping Type
	local

BSW Module	BSW Context	
CanTp	CanTp/CanTpConfig/CanTpChannel/CanTpRxNSdu	
BSW Parameter		**BSW Type**
CanTpRxNSduId		EcucIntegerParamDef
BSW Description		
Unique identifier user by the upper layer to call CanTp_CancelReceive, CanTp_ChangeParameter and CanTp_ReadParameter.		
M2 Template	**M2 Description**	
SystemTemplate	To describe a frames identifier on the communication system, usualy with a fixed identifierValue.	
M2 Parameter		
Fibex::Fibex4Can::CanCommunication::CanFrameTriggering.identifier		
Mapping Rule		**Mapping Type**
Id described by the CanId in the FrameTriggering.		full

BSW Module	BSW Context	
CanTp	CanTp/CanTpConfig/CanTpChannel/CanTpRxNSdu	
BSW Parameter		**BSW Type**
CanTpRxNSduRef		EcucReferenceDef
BSW Description		
Reference to a Pdu in the COM-Stack.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
CanTp	CanTp/CanTpConfig/CanTpChannel/CanTpRxNSdu	
BSW Parameter		**BSW Type**
CanTpRxPaddingActivation		EcucEnumerationParamDef
BSW Description		
Defines if the receive frame uses padding or not. Definition of enumeration values: CanTpOn: The N-PDU received uses padding for SF, FC and the last CF. (N-PDU length is always 8 bytes) CanTpOff: The N-PDU received does not use padding for SF, CF and the last CF. (N-PDU length is dynamic)		
M2 Template	**M2 Description**	
SystemTemplate	This specifies wheter or not Sfs, FCs and the last CF shall be padded to 8 bytes length in case it contains less payload.	
M2 Parameter		
TransportProtocols::CanTpConnection.paddingActivation		
Mapping Rule		**Mapping Type**

1:1 mapping	full

BSW Module	**BSW Context**	
CanTp	CanTp/CanTpConfig/CanTpChannel/CanTpRxNSdu	
BSW Parameter		**BSW Type**
CanTpRxTaType		EcucEnumerationParamDef
BSW Description		
Declares the communication type of this Rx N-SDU.		
M2 Template	**M2 Description**	
SystemTemplate	Network Target Address type.	
M2 Parameter		
TransportProtocols::CanTpConnection.taType		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
CanTp	CanTp/CanTpConfig/CanTpChannel/CanTpRxNSdu	
BSW Parameter		**BSW Type**
CanTpRxWftMax		EcucIntegerParamDef
BSW Description		
This parameter indicates how many Flow Control wait N-PDUs can be consecutively transmitted by the receiver. It is local to the node and is not transmitted inside the FC protocol data unit. CanTpRxWftMax is used to avoid sender nodes being potentially hooked-up in case of a temporarily reception inability on the part of the receiver nodes, whereby the sender could be waiting continuously.		
M2 Template	**M2 Description**	
SystemTemplate	This attribute defines the maximum number of flow control PDUs that can be consecutively be transmitted by a receiver.	
M2 Parameter		
TransportProtocols::CanTpNode.maxFcWait		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
CanTp	CanTp/CanTpConfig/CanTpChannel/CanTpRxNSdu	
BSW Parameter		**BSW Type**
CanTpSTmin		EcucFloatParamDef
BSW Description		
Sets the duration of the minimum time the CanTp sender shall wait between the transmissions of two CF N-PDUs. For further details on this parameter value see ISO 15765-2 specification.		
M2 Template	**M2 Description**	
SystemTemplate	This parameter defines the minimum amount of time (in seconds) between two succeeding CFs.	
M2 Parameter		
TransportProtocols::CanTpNode.stMin		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
CanTp	CanTp/CanTpConfig/CanTpChannel/CanTpRxNSdu	
BSW Parameter		**BSW Type**
CanTpTxFcNPdu		EcucParamConfContainerDef
BSW Description		
Used for grouping of the ID of a PDU and the Reference to a PDU.		
M2 Template	**M2 Description**	
SystemTemplate	Reference to the Flow Control NPdu.	
M2 Parameter		
TransportProtocols::CanTpConnection.flowControlPdu		
Mapping Rule		**Mapping Type**
Create container if the CanTpConnection contains a reference to a FlowControl NPdu that is received by the regarded ECU.		full

BSW Module	BSW Context	
CanTp	CanTp/CanTpConfig/CanTpChannel/CanTpRxNSdu/CanTpTxFcNPdu	
BSW Parameter		**BSW Type**
CanTpTxFcNPduConfirmationPduId		EcucIntegerParamDef
BSW Description		
Handle Id to be used by the CanIf to confirm the transmission of the CanTpTxFcNPdu to the CanIf module.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
CanTp	CanTp/CanTpConfig/CanTpChannel/CanTpRxNSdu/CanTpTxFcNPdu	
BSW Parameter		**BSW Type**
CanTpTxFcNPduRef		EcucReferenceDef
BSW Description		
Reference to a Pdu in the COM-Stack.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
CanTp	CanTp/CanTpConfig/CanTpChannel	
BSW Parameter		**BSW Type**
CanTpTxNSdu		EcucParamConfContainerDef
BSW Description		
The following parameters needs to be configured for each CAN N-SDU that the CanTp module transmits via the CanTpChannel.		
M2 Template	**M2 Description**	
SystemTemplate	Reference to the IPdu that is segmented by the Transport Protocol.	

M2 Parameter	
TransportProtocols::CanTpConnection.tpSdu	
Mapping Rule	**Mapping Type**
Create container for each existing CanTpConnection that contains a reference to an N-SDU that is transmitted.	full

BSW Module	BSW Context		
CanTp	CanTp/CanTpConfig/CanTpChannel/CanTpTxNSdu		
BSW Parameter		**BSW Type**	
CanTpNAe		EcucParamConfContainerDef	
BSW Description			
Contains the parameters needed to configure each RxNSdu or TxNSdu with CanTpAddressingFormat set to CanTpMixed.			
M2 Template	**M2 Description**		
SystemTemplate	Declares which communication addressing mode is supported.		
M2 Parameter			
TransportProtocols::CanTpConnection.addressingFormat			
Mapping Rule			**Mapping Type**
Create container if addressingFormat is set to "mixed".			full

BSW Module	BSW Context		
CanTp	CanTp/CanTpConfig/CanTpChannel/CanTpTxNSdu/CanTpNAe		
BSW Parameter		**BSW Type**	
CanTpNAe		EcucIntegerParamDef	
BSW Description			
If an RxNsdu or a TxNsdu is configured for mixed addressing format, this parameter contains the transport protocol address extension value.			
M2 Template	**M2 Description**		
SystemTemplate	An ECUs TP address on the referenced channel. This represents the diagnostic Address.		
M2 Parameter			
TransportProtocols::TPAddress.tpAddressExtensionValue			
Mapping Rule			**Mapping Type**
The CanTPConnection contains a reference to the SDU and a relation to the Tp Node that contains the TpAddressExtension.			full

BSW Module	BSW Context		
CanTp	CanTp/CanTpConfig/CanTpChannel/CanTpTxNSdu		
BSW Parameter		**BSW Type**	
CanTpNSa		EcucParamConfContainerDef	
BSW Description			
Contains the parameters needed to configure each RxNSdu or TxNSdu with CanTpAddressingFormat set to CanTpExtended.			
M2 Template	**M2 Description**		
SystemTemplate	Declares which communication addressing mode is supported.		
M2 Parameter			
TransportProtocols::CanTpConnection.addressingFormat			
Mapping Rule			**Mapping Type**
Create container if addressingFormat is set to "extended".			full

BSW Module	BSW Context	
CanTp	CanTp/CanTpConfig/CanTpChannel/CanTpTxNSdu/CanTpNSa	
BSW Parameter		**BSW Type**
CanTpNSa		EcucIntegerParamDef
BSW Description		
If an RxNSdu or a TxNSdu is configured for extended addressing format, this parameter contains the transport protocol source address's value.		
M2 Template	**M2 Description**	
System Template	An ECUs TP address on the referenced channel. This represents the diagnostic Address.	
M2 Parameter		
TransportProtocols::TPAddress.tpAddress		
Mapping Rule		**Mapping Type**
The CanTPConnection contains a reference to the SDU and a relation to the Tp Node that contains the TpAddress.		full

BSW Module	BSW Context	
CanTp	CanTp/CanTpConfig/CanTpChannel/CanTpTxNSdu	
BSW Parameter		**BSW Type**
CanTpNTa		EcucParamConfContainerDef
BSW Description		
The following parameters need to be configured for each RxNsdu or TxNsdu with the CanTpAddressingFormat set to CanTpExtended.		
M2 Template	**M2 Description**	
SystemTemplate	Declares which communication addressing mode is supported.	
M2 Parameter		
TransportProtocols::CanTpConnection.addressingFormat		
Mapping Rule		**Mapping Type**
Create container if addressingFormat is set to "extended".		full

BSW Module	BSW Context	
CanTp	CanTp/CanTpConfig/CanTpChannel/CanTpTxNSdu/CanTpNTa	
BSW Parameter		**BSW Type**
CanTpNTa		EcucIntegerParamDef
BSW Description		
If an RxNsdu or a TxNsdu is configured for extended addressing format, this parameter contains the transport protocol target address's value.		
M2 Template	**M2 Description**	
SystemTemplate	An ECUs TP address on the referenced channel. This represents the diagnostic Address.	
M2 Parameter		
TransportProtocols::TPAddress.tpAddress		
Mapping Rule		**Mapping Type**
The CanTPConnection contains a reference to the SDU and a relation to the Tp Node that contains the TpAddress.		full

BSW Module	BSW Context	
CanTp	CanTp/CanTpConfig/CanTpChannel/CanTpTxNSdu	
BSW Parameter		**BSW Type**
CanTpNas		EcucFloatParamDef

BSW Description		
Value in second of the N_As timeout. N_As is the time for transmission of a CAN frame (any N_PDU) on the part of the sender.		
M2 Template	**M2 Description**	
SystemTemplate	Timeout between the PDU transmit request for the first PDU of the group used in the current connection of the Transport Layer to the Can Interface and the corresponding confirmation of the Can Interface on the sender side	
M2 Parameter		
TransportProtocols::CanTpNode.timeoutAs		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
CanTp	CanTp/CanTpConfig/CanTpChannel/CanTpTxNSdu	
BSW Parameter		**BSW Type**
CanTpNbs		EcucFloatParamDef
BSW Description		
Value in seconds of the N_Bs timeout. N_Bs is the time of transmission until reception of the next Flow Control N_PDU.		
M2 Template	**M2 Description**	
SystemTemplate	This parameter defines the timout for waiting for an FC or AF on the sender side in an 1:1 connection. Specified in seconds.	
M2 Parameter		
TransportProtocols::CanTpConnection.timeoutBs		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
CanTp	CanTp/CanTpConfig/CanTpChannel/CanTpTxNSdu	
BSW Parameter		**BSW Type**
CanTpNcs		EcucFloatParamDef
BSW Description		
Value in seconds of the performance requirement of (N_Cs + N_As). N_Cs is the time which elapses between the transmit request of a CF N-PDU until the transmit request of the next CF N-PDU.		
M2 Template	**M2 Description**	
SystemTemplate	timeoutCs is the time which elapses between the transmit request of a CF N-PDU until the transmit request of the next CF N-PDU.	
M2 Parameter		
TransportProtocols::CanTpConnection.timeoutCs		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
CanTp	CanTp/CanTpConfig/CanTpChannel/CanTpTxNSdu	
BSW Parameter		**BSW Type**
CanTpRxFcNPdu		EcucParamConfContainerDef
BSW Description		
Used for grouping of the ID of a PDU and the Reference to a PDU.		
M2 Template	**M2 Description**	
SystemTemplate	Reference to the Flow Control NPdu.	

M2 Parameter	
TransportProtocols::CanTpConnection.flowControlPdu	
Mapping Rule	**Mapping Type**
Create container if the CanTpConnection contains a reference to a FlowControl NPdu that is received by the regarded ECU.	full

BSW Module	BSW Context	
CanTp	CanTp/CanTpConfig/CanTpChannel/CanTpTxNSdu/CanTpRxFcNPdu	
BSW Parameter		**BSW Type**
CanTpRxFcNPduId		EcucIntegerParamDef
BSW Description		
N-PDU identifier attached to the FC N-PDU of this TxNsdu identified by CanTpTxNSduId. Each TxNsdu identifier is linked to one Rx FC N-PDU identifier only. However, in the case of extended addressing format, the same FC N-PDU identifier can be used for several N-SDU identifiers. The distinction is made by means of the N_TA value (first data byte of FC frames).		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
CanTp	CanTp/CanTpConfig/CanTpChannel/CanTpTxNSdu/CanTpRxFcNPdu	
BSW Parameter		**BSW Type**
CanTpRxFcNPduRef		EcucReferenceDef
BSW Description		
Reference to a Pdu in the COM-Stack.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
CanTp	CanTp/CanTpConfig/CanTpChannel/CanTpTxNSdu	
BSW Parameter		**BSW Type**
CanTpTc		EcucBooleanParamDef
BSW Description		
switch for enabling Transmit Cancellation.		
M2 Template	**M2 Description**	
SystemTemplate	With this switch Transmit Cancellation can be turned on or off for this channel.	
M2 Parameter		
TransportProtocols::CanTpConnection.transmitCancellation		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
CanTp	CanTp/CanTpConfig/CanTpChannel/CanTpTxNSdu	
BSW Parameter		**BSW Type**
CanTpTxAddressingFormat		EcucEnumerationParamDef
BSW Description		
Declares which communication addressing format is supported for this TxNsdu. Definition of Enumeration values: CanTpStandard to use normal addressing format. CanTpExtended to use extended addressing format (the N_TA container of this TxNsdu will be used). CanTpMixed to use mixed addressing format (the N_AE container of this TxNsdu will be used).		
M2 Template	**M2 Description**	
SystemTemplate	Declares which communication addressing mode is supported.	
M2 Parameter		
TransportProtocols::CanTpConnection.addressingFormat		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
CanTp	CanTp/CanTpConfig/CanTpChannel/CanTpTxNSdu	
BSW Parameter		**BSW Type**
CanTpTxDl		EcucIntegerParamDef
BSW Description		
Data Length Code of this TxNsdu. In case of variable length message, this value indicates the minimum data length.		
M2 Template	**M2 Description**	
SystemTemplate	Maximum Pdu length in bits. In case of dynamic length IPdus (containing a dynamical length signal), this value indicates the minimum data length.	
M2 Parameter		
Pdu.Length		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
CanTp	CanTp/CanTpConfig/CanTpChannel/CanTpTxNSdu	
BSW Parameter		**BSW Type**
CanTpTxNPdu		EcucParamConfContainerDef
BSW Description		
Used for grouping of the ID of a PDU and the Reference to a PDU.		
M2 Template	**M2 Description**	
SystemTemplate	Reference to an Data NPdu.	
M2 Parameter		
TransportProtocols::CanTpConnection dataPdu		
Mapping Rule		**Mapping Type**
Create container if the CanTpConnection contains a reference to a DataNpdu that is received by the regarded ECU.		full

BSW Module	BSW Context	
CanTp	CanTp/CanTpConfig/CanTpChannel/CanTpTxNSdu/CanTpTxNPdu	
BSW Parameter		**BSW Type**

CanTpTxNPduConfirmationPduId		EcucIntegerParamDef	
BSW Description			
Handle Id to be used by the CanIf to confirm the transmission of the CanTpTxNPdu to the CanIf module.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	**BSW Context**		
CanTp	CanTp/CanTpConfig/CanTpChannel/CanTpTxNSdu/CanTpTxNPdu		
BSW Parameter		**BSW Type**	
CanTpTxNPduRef		EcucReferenceDef	
BSW Description			
Reference to a Pdu in the COM-Stack.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	**BSW Context**		
CanTp	CanTp/CanTpConfig/CanTpChannel/CanTpTxNSdu		
BSW Parameter		**BSW Type**	
CanTpTxNSduId		EcucIntegerParamDef	
BSW Description			
Unique identifier to a structure that contains all useful information to process the transmission of a TxNsdu.			
M2 Template	**M2 Description**		
SystemTemplate	To describe a frames identifier on the communication system, usualy with a fixed identifierValue.		
M2 Parameter			
Fibex::Fibex4Can::CanCommunication::CanFrameTriggering.identifier			
Mapping Rule			**Mapping Type**
Id described by the CanId in the FrameTriggering.			full

BSW Module	**BSW Context**		
CanTp	CanTp/CanTpConfig/CanTpChannel/CanTpTxNSdu		
BSW Parameter		**BSW Type**	
CanTpTxNSduRef		EcucReferenceDef	
BSW Description			
Reference to a Pdu in the COM-Stack.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**

	local

BSW Module	BSW Context		
CanTp	CanTp/CanTpConfig/CanTpChannel/CanTpTxNSdu		
BSW Parameter		**BSW Type**	
CanTpTxPaddingActivation		EcucEnumerationParamDef	
BSW Description			
Defines if the transmit frame use padding or not. Definition of Enumeration values: CanTpOn The transmit N-PDU uses padding for SF, FC and the last CF. (N-PDU length is always 8 bytes) CanTpOff The transmit N-PDU does not use padding for SF, CF and the last CF. (N-PDU length is dynamic)			
M2 Template	**M2 Description**		
SystemTemplate	This specifies wheter or not Sfs, FCs and the last CF shall be padded to 8 bytes length in case it contains less payload.		
M2 Parameter			
TransportProtocols::CanTpConnection.paddingActivation			
Mapping Rule			**Mapping Type**
1:1 mapping			full

BSW Module	BSW Context		
CanTp	CanTp/CanTpConfig/CanTpChannel/CanTpTxNSdu		
BSW Parameter		**BSW Type**	
CanTpTxTaType		EcucEnumerationParamDef	
BSW Description			
Declares the communication type of this TxNsdu. Enumeration values: CanTpPhysical. Used for 1:1 communication. CanTpFunctional. Used for 1:n communication.			
M2 Template	**M2 Description**		
SystemTemplate	Network Target Address type.		
M2 Parameter			
TransportProtocols::CanTpConnection.taType			
Mapping Rule			**Mapping Type**
1:1 mapping			full

BSW Module	BSW Context	
CanTp	CanTp/CanTpConfig	
BSW Parameter		**BSW Type**
CanTpMainFunctionPeriod		EcucFloatParamDef
BSW Description		
Allow to configure the time for the MainFunction (as float in seconds). Please note: This period shall be the same as call cycle time of the periodic task were CanTp Main function is called.		
M2 Template	**M2 Description**	
SystemTemplate	The period between successive calls to the Main Function of the ASR TP. Specified in seconds.	

M2 Parameter	
TransportProtocols::TpEcu.cycleTimeMainFunction	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	**BSW Context**	
CanTp	CanTp	
BSW Parameter		**BSW Type**
CanTpGeneral		EcucParamConfContainerDef
BSW Description		
This container contains the general configuration parameters of the CanTp module.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
CanTp	CanTp/CanTpGeneral	
BSW Parameter		**BSW Type**
CanTpChangeParameterApi		EcucBooleanParamDef
BSW Description		
This parameter, if set to true, enables the CanTp_ChangeParameterRequest Api for this Module.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
CanTp	CanTp/CanTpGeneral	
BSW Parameter		**BSW Type**
CanTpDevErrorDetect		EcucBooleanParamDef
BSW Description		
Switches the Development Error Detection and Notification ON or OFF		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
CanTp	CanTp/CanTpGeneral	
BSW Parameter		**BSW Type**
CanTpPaddingByte		EcucIntegerParamDef
BSW Description		

Used for the initialization of unused bytes with a certain value		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
CanTp	CanTp/CanTpGeneral	
BSW Parameter		**BSW Type**
CanTpReadParameterApi		EcucBooleanParamDef
BSW Description		
This parameter, if set to true, enables the CanTp_ReadParameterApi for this module.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
CanTp	CanTp/CanTpGeneral	
BSW Parameter		**BSW Type**
CanTpVersionInfoApi		EcucBooleanParamDef
BSW Description		
The function CanTp_GetVersionInfo is configurable (On/Off) by this configuration parameter.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

D.5 CanTrcv Mapping

BSW Module	**BSW Context**	
CanTrcv	CanTrcv	
BSW Parameter		**BSW Type**
CanTrcvConfigSet		EcucParamConfContainerDef
BSW Description		
This is the multiple configuration set container for CAN Transceiver.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**

BSW Module	BSW Context	
CanTrcv	CanTrcv/CanTrcvConfigSet	
BSW Parameter		**BSW Type**
CanTrcvChannel		EcucParamConfContainerDef
BSW Description		
Container gives CAN transceiver driver information about a single CAN transceiver (channel).		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**

BSW Module	BSW Context	
CanTrcv	CanTrcv/CanTrcvConfigSet/CanTrcvChannel	
BSW Parameter		**BSW Type**
CanTrcvAccess		EcucChoiceContainerDef
BSW Description		
Container gives CanTrcv Driver information about access to a single CAN transceiver.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**

BSW Module	BSW Context	
CanTrcv	CanTrcv/CanTrcvConfigSet/CanTrcvChannel	
BSW Parameter		**BSW Type**
CanTrcvChannelId		EcucIntegerParamDef
BSW Description		
Unique identifier of the CAN Transceiver Channel.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**

BSW Module	BSW Context	
CanTrcv	CanTrcv/CanTrcvConfigSet/CanTrcvChannel	
BSW Parameter		**BSW Type**
CanTrcvChannelUsed		EcucBooleanParamDef
BSW Description		
Shall the related CAN transceiver channel be used?		
M2 Template	**M2 Description**	
M2 Parameter		

Mapping Rule	Mapping Type

BSW Module	BSW Context	
CanTrcv	CanTrcv/CanTrcvConfigSet/CanTrcvChannel	
BSW Parameter		**BSW Type**
CanTrcvControlsPowerSupply		EcucBooleanParamDef
BSW Description		
Is ECU power supply controlled by this transceiver? TRUE = Controlled by transceiver. FALSE = Not controlled by transceiver.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**

BSW Module	BSW Context	
CanTrcv	CanTrcv/CanTrcvConfigSet/CanTrcvChannel	
BSW Parameter		**BSW Type**
CanTrcvHwPnSupport		EcucBooleanParamDef
BSW Description		
Indicates whether the HW supports the selective wake-up function TRUE = Selective wakeup feature is supported by the transceiver FALSE = Selective wakeup feature is supported by the transceiver.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**

BSW Module	BSW Context	
CanTrcv	CanTrcv/CanTrcvConfigSet/CanTrcvChannel	
BSW Parameter		**BSW Type**
CanTrcvInitState		EcucEnumerationParamDef
BSW Description		
State of CAN transceiver after call to CanTrcv_Init.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**

BSW Module	BSW Context
CanTrcv	CanTrcv/CanTrcvConfigSet/CanTrcvChannel

BSW Parameter	**BSW Type**
CanTrcvMaxBaudrate	EcucIntegerParamDef
BSW Description	
Max baudrate for transceiver hardware type. Only used for validation purposes. Value shall be configured by configuration tool based on transceiver hardware type.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**

BSW Module	**BSW Context**
CanTrcv	CanTrcv/CanTrcvConfigSet/CanTrcvChannel
BSW Parameter	**BSW Type**
CanTrcvPartialNetwork	EcucParamConfContainerDef
BSW Description	
Container gives CAN transceiver driver information about the configuration of Partial Networking functionality.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**
CanTrcv	CanTrcv/CanTrcvConfigSet/CanTrcvChannel/CanTrcvPartialNetwork
BSW Parameter	**BSW Type**
CanTrcvBaudRate	EcucIntegerParamDef
BSW Description	
Indicates the CAN Bus communication baud rate in kbps.	
M2 Template	**M2 Description**
SystemTemplate	
M2 Parameter	
CoreTopology::CommunicationCluster.speed	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	**BSW Context**
CanTrcv	CanTrcv/CanTrcvConfigSet/CanTrcvChannel/CanTrcvPartialNetwork
BSW Parameter	**BSW Type**
CanTrcvBusErrFlag	EcucBooleanParamDef
BSW Description	
Indicates if the Bus Error (BUSERR) flag is managed by the BSW. This flag is set if a bus failure is detected by the transceiver. TRUE = Supported by transceiver and managed by BSW. FALSE = Not managed by BSW.	
M2 Template	**M2 Description**

M2 Parameter	
Mapping Rule	**Mapping Type**

BSW Module	**BSW Context**	
CanTrcv	CanTrcv/CanTrcvConfigSet/CanTrcvChannel/CanTrcvPartialNetwork	
BSW Parameter		**BSW Type**
CanTrcvPnCanIdIsExtended		EcucBooleanParamDef
BSW Description		
Indicates whether extended or standard ID is used. TRUE = Extended Can identifier is used. FALSE = Standard Can identifier is used		
M2 Template	**M2 Description**	
SystemTemplate		
M2 Parameter		
CanTopology::CanCommunicationConnector.pncWakeupCanIdExtended		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
CanTrcv	CanTrcv/CanTrcvConfigSet/CanTrcvChannel/CanTrcvPartialNetwork	
BSW Parameter		**BSW Type**
CanTrcvPnEnabled		EcucBooleanParamDef
BSW Description		
Indicates whether the selective wake-up function is enabled or disabled in HW. TRUE = Selective wakeup feature is enabled in the transceiver hardware FALSE = Selective wakeup feature is disabled in the transceiver hardware		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**

BSW Module	**BSW Context**	
CanTrcv	CanTrcv/CanTrcvConfigSet/CanTrcvChannel/CanTrcvPartialNetwork	
BSW Parameter		**BSW Type**
CanTrcvPnFrameCanId		EcucIntegerParamDef
BSW Description		
CAN ID of the Wake-up Frame (WUF).		
M2 Template	**M2 Description**	
SystemTemplate		
M2 Parameter		
CanTopology::CanCommunicationConnector.pncWakeupCanId		
Mapping Rule		**Mapping Type**
1:1 mapping		full

— AUTOSAR CONFIDENTIAL —

BSW Module	BSW Context	
CanTrcv	CanTrcv/CanTrcvConfigSet/CanTrcvChannel/CanTrcvPartialNetwork	
BSW Parameter		**BSW Type**
CanTrcvPnFrameCanIdMask		EcucIntegerParamDef
BSW Description		
ID Mask for the selective activation of the transceiver. It is used to enableFrame Wake-up (WUF) on a group of IDs.		
M2 Template	**M2 Description**	
SystemTemplate		
M2 Parameter		
CanTopology::CanCommunicationConnector.pncWakeupCanIdMask		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
CanTrcv	CanTrcv/CanTrcvConfigSet/CanTrcvChannel/CanTrcvPartialNetwork	
BSW Parameter		**BSW Type**
CanTrcvPnFrameDataMaskSpec		EcucParamConfContainerDef
BSW Description		
Defines data payload mask to be used on the received payload in order to determine if the transceiver must be woken up by the received Wake-up Frame (WUF).		
M2 Template	**M2 Description**	
SystemTemplate		
M2 Parameter		
CanTopology::CanCommunicationConnector.pncWakeupDataMask		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
CanTrcv	CanTrcv/CanTrcvConfigSet/CanTrcvChannel/CanTrcvPartialNetwork/CanTrcvPnFrameDataMaskSpec	
BSW Parameter		**BSW Type**
CanTrcvPnFrameDataMask		EcucIntegerParamDef
BSW Description		
Defines the n byte (Byte0 = LSB) of the data payload mask to be used on the received payload in order to determine if the transceiver must be woken up by the received Wake-up Frame (WUF).		
M2 Template	**M2 Description**	
SystemTemplate		
M2 Parameter		
CanTopology::CanCommunicationConnector.pncWakeupDataMask		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
CanTrcv	CanTrcv/CanTrcvConfigSet/CanTrcvChannel/CanTrcvPartialNetwork/CanTrcvPnFrameDataMaskSpec	
BSW Parameter		**BSW Type**
CanTrcvPnFrameDataMaskIndex		EcucIntegerParamDef
BSW Description		
holds the position n in frame of the mask-part		

M2 Template	M2 Description	
SystemTemplate		
M2 Parameter		
CanTopology::CanCommunicationConnector.pncWakeupDataMask		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context		
CanTrcv	CanTrcv/CanTrcvConfigSet/CanTrcvChannel/CanTrcvPartialNetwork		
BSW Parameter		**BSW Type**	
CanTrcvPnFrameDlc		EcucIntegerParamDef	
BSW Description			
Data Length of the Wake-up Frame (WUF).			
M2 Template	**M2 Description**		
SystemTemplate			
M2 Parameter			
CanTopology::CanCommunicationConnector.pncWakeupDlc			
Mapping Rule			**Mapping Type**
1:1 mapping			full

BSW Module	BSW Context		
CanTrcv	CanTrcv/CanTrcvConfigSet/CanTrcvChannel/CanTrcvPartialNetwork		
BSW Parameter		**BSW Type**	
CanTrcvPowerOnFlag		EcucBooleanParamDef	
BSW Description			
Description: Indicates if the Power On Reset (POR) flag is available and is managed by the transceiver. TRUE = Supported by Hardware. FALSE = Not supported by Hardware			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**

BSW Module	BSW Context	
CanTrcv	CanTrcv/CanTrcvConfigSet/CanTrcvChannel	
BSW Parameter		**BSW Type**
CanTrcvWakeupByBusUsed		EcucBooleanParamDef
BSW Description		
Is wake up by bus supported? If CAN transceiver hardware does not support wake up by bus value is always FALSE. If CAN transceiver hardware supports wake up by bus value is TRUE or FALSE depending whether it is used or not. TRUE = Is used. FALSE = Is not used.		
M2 Template	**M2 Description**	
M2 Parameter		

Mapping Rule	**Mapping Type**

BSW Module	**BSW Context**	
CanTrcv	CanTrcv/CanTrcvConfigSet/CanTrcvChannel	
BSW Parameter		**BSW Type**
CanTrcvWakeupSourceRef		EcucReferenceDef
BSW Description		
Reference to a wakeup source in the EcuM configuration. This reference is only needed if CanTrcvWakeupByBusUsed is true.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**

BSW Module	**BSW Context**	
CanTrcv	CanTrcv/CanTrcvConfigSet	
BSW Parameter		**BSW Type**
CanTrcvSPICommRetries		EcucIntegerParamDef
BSW Description		
Indicates the maximum number of communication retries in case of a failed SPI communication (applies both to timed out communication and to errors/NACK in the response data). If configured value is ’0’, no retry is allowed (communication is expected to succeed at first try).		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**

BSW Module	**BSW Context**	
CanTrcv	CanTrcv/CanTrcvConfigSet	
BSW Parameter		**BSW Type**
CanTrcvSPICommTimeout		EcucIntegerParamDef
BSW Description		
Indicates the maximum time allowed to the CanTrcv for replying (either positively or negatively) to a SPI command. Timeout is configured in milliseconds. Timeout value of ’0’ means that no specific timeout is to be used by CanTrcv and the communication is executed at the best of the SPI HW capacity.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**

— AUTOSAR CONFIDENTIAL —

BSW Module	**BSW Context**
CanTrcv	CanTrcv
BSW Parameter	**BSW Type**
CanTrcvGeneral	EcucParamConfContainerDef
BSW Description	
Container gives CAN transceiver driver basic information.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**

BSW Module	**BSW Context**
CanTrcv	CanTrcv/CanTrcvGeneral
BSW Parameter	**BSW Type**
CanTrcvDevErrorDetect	EcucBooleanParamDef
BSW Description	
Switches development error detection and notification on and off. If switched on, #define CANTRCV_DEV _ERROR_DETECT ON shall be generated. If switched off, #define CANTRCV_DEV_ERROR _DETECT OFF shall be generated. Define shall be part of file CanTrcv_Cfg.h.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**

BSW Module	**BSW Context**
CanTrcv	CanTrcv/CanTrcvGeneral
BSW Parameter	**BSW Type**
CanTrcvGetVersionInfo	EcucBooleanParamDef
BSW Description	
Switches version information API on and off. If switched off, function need not be present in compiled code.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**

BSW Module	**BSW Context**
CanTrcv	CanTrcv/CanTrcvGeneral
BSW Parameter	**BSW Type**
CanTrcvSPICommRetries	EcucIntegerParamDef
BSW Description	

Indicates the maximal number of communication retries in case of failed SPI communication (applies both to timed out communication and to errors/NACK in the response data). (0 ... 255 times, 0 means no retry allowed, communication must succeed at first try)		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**

BSW Module	**BSW Context**	
CanTrcv	CanTrcv/CanTrcvGeneral	
BSW Parameter		**BSW Type**
CanTrcvSPICommTimeout		EcucIntegerParamDef
BSW Description		
Indicates the maximal time allowed to the Transceiver in order to reply (either positively or negatively) to a SPI command. (value in ms, 0ms means no specific timeout is to be used, communication is executed at the best of the SPI HW capacity)		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**

BSW Module	**BSW Context**	
CanTrcv	CanTrcv/CanTrcvGeneral	
BSW Parameter		**BSW Type**
CanTrcvWaitCount		EcucIntegerParamDef
BSW Description		
Indicates the number of wait states to change the transceiver operation mode. Transceiver hardware may need wait states for some transitions.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**

BSW Module	**BSW Context**	
CanTrcv	CanTrcv/CanTrcvGeneral	
BSW Parameter		**BSW Type**
CanTrcvWakeUpSupport		EcucEnumerationParamDef
BSW Description		
Informs whether wake up is supported by polling or not supported. In case no wake up is supported by the hardware, setting has to be NOT_SUPPORTED. Only in the case of wake up supported by polling, function CanTrcv_MainFunction has to be present and to be invoked by the scheduler.		

M2 Template	M2 Description	
M2 Parameter		
Mapping Rule		**Mapping Type**

D.6 FlexRay Driver Mapping

BSW Module	BSW Context		
Fr	Fr		
BSW Parameter		**BSW Type**	
FrGeneral		EcucParamConfContainerDef	
BSW Description			
General configuration (parameters) of the FlexRay Driver module.			
M2 Template	**M2 Description**		
System Template	FlexRay specific attributes to the physicalCluster		
M2 Parameter			
Fibex4FlexRay::FlexRayCluster			
Mapping Rule			**Mapping Type**
Container must be created if the ECU is connected to a FlexRay Cluster			full

BSW Module	BSW Context		
Fr	Fr/FrGeneral		
BSW Parameter		**BSW Type**	
FrBufferReconfig		EcucBooleanParamDef	
BSW Description			
Enables or disables buffer reconfiguration at runtime.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	BSW Context		
Fr	Fr/FrGeneral		
BSW Parameter		**BSW Type**	
FrCtrlTestCount		EcucIntegerParamDef	
BSW Description			
Maxmimum number of iterations the FlexRay controller hardware test is performed during controller initialization.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

— AUTOSAR CONFIDENTIAL —

BSW Module	BSW Context	
Fr	Fr/FrGeneral	
BSW Parameter		**BSW Type**
FrDevErrorDetect		EcucBooleanParamDef
BSW Description		
Switches the Development Error Detection and Notification on or off. true: Development Error Detection and Notification enabled. false: Development Error Detection and Notification disabled.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
Fr	Fr/FrGeneral	
BSW Parameter		**BSW Type**
FrIndex		EcucIntegerParamDef
BSW Description		
Specifies the InstanceId of this module instance. If only one instance is present it shall have the Id 0.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
Fr	Fr/FrGeneral	
BSW Parameter		**BSW Type**
FrNumCtrlSupported		EcucIntegerParamDef
BSW Description		
Determines the maximum number of communication controllers that the driver supports.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
Fr	Fr/FrGeneral	
BSW Parameter		**BSW Type**
FrRxStringentCheck		EcucBooleanParamDef
BSW Description		
If stringent check is enabled (true), received frames are only accepted if no slot status error occured.		

M2 Template	M2 Description	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context		
Fr	Fr/FrGeneral		
BSW Parameter		**BSW Type**	
FrRxStringentLengthCheck		EcucBooleanParamDef	
BSW Description			
If stringent check is enabled (true), received frames are only accepted the received payload length matches the configured payload length.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	BSW Context		
Fr	Fr/FrGeneral		
BSW Parameter		**BSW Type**	
FrVersionInfoApi		EcucBooleanParamDef	
BSW Description			
Enables/disables the existence of the Fr_GetVersionInfo API.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	BSW Context		
Fr	Fr		
BSW Parameter		**BSW Type**	
FrMultipleConfiguration		EcucParamConfContainerDef	
BSW Description			
Configuration of the individual controllers.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**

BSW Module	BSW Context
Fr	Fr/FrMultipleConfiguration

BSW Parameter		BSW Type
FrController		EcucParamConfContainerDef
BSW Description		
Configuration of the individual controller.		
M2 Template	**M2 Description**	
System Template	The communication controller is a dedicated hardware device by means of which hosts are sending frames to and receiving frames from the communication medium.	
M2 Parameter		
Fibex:FibexCore::Topology::EcuInstance::CommunicationController		
Mapping Rule		**Mapping Type**
Container must be created if the ECU contains a FlexRay communication controller that is connected to the regarded communication cluster.		full

BSW Module	BSW Context	
Fr	Fr/FrMultipleConfiguration/FrController	
BSW Parameter		**BSW Type**
FrAbsoluteTimer		EcucParamConfContainerDef
BSW Description		
Specifies the absolute timer configuration parameters of the Fr.		
M2 Template	**M2 Description**	
ECU Resource Template		
M2 Parameter		
Mapping Rule		**Mapping Type**
		unknown

BSW Module	BSW Context	
Fr	Fr/FrMultipleConfiguration/FrController/FrAbsoluteTimer	
BSW Parameter		**BSW Type**
FrAbsTimerIdx		EcucIntegerParamDef
BSW Description		
Contains the index of an absolute timer contained in Fr on a certain FlexRay CC.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
Fr	Fr/FrMultipleConfiguration/FrController	
BSW Parameter		**BSW Type**
FrControllerDemEventParameterRefs		EcucParamConfContainerDef
BSW Description		
Container for the references to DemEventParameter elements which shall be invoked using the API Dem_ReportErrorStatus API in case the corresponding error occurs. The EventId is taken from the referenced DemEventParameter's DemEventId value. The standardized errors are provided in the container and can be extended by vendor specific error references.		

M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**
Fr	Fr/FrMultipleConfiguration/FrController/FrControllerDemEventParameterRefs
BSW Parameter	**BSW Type**
FrDemCtrlTestResultRef	EcucSymbolicNameReferenceDef
BSW Description	
Reference to DEM event Id that is reported for FlexRay controller hardware test failure. If this parameter is not configured, no event reporting happens.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**
Fr	Fr/FrMultipleConfiguration/FrController
BSW Parameter	**BSW Type**
FrCtrlIdx	EcucIntegerParamDef
BSW Description	
Determines index of CC within Fr.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**
Fr	Fr/FrMultipleConfiguration/FrController
BSW Parameter	**BSW Type**
FrFifo	EcucParamConfContainerDef
BSW Description	
One First In First Out (FIFO) queued receive structure, defining the admittance criteria to the FIFO, and mandating the ability to admit messages into the FIFO based on Message Id filtering criteria.	
M2 Template	**M2 Description**
SystemTemplate	One First In First Out (FIFO) queued receive structure, defining the admittance criteria to the FIFO, and mandating the ability to admit messages into the FIFO based on Message Id filtering criteria.
M2 Parameter	
Fibex::Fibex4FlexRay::FlexRayFifoConfiguration	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	BSW Context	
Fr	Fr/FrMultipleConfiguration/FrController/FrFifo	
BSW Parameter		**BSW Type**
FrAdmitWithoutMessageId		EcucBooleanParamDef
BSW Description		
Determines whether or not frames received in the dynamic segment that don't contain a message ID will be admitted into the FIFO.		
M2 Template	**M2 Description**	
SystemTemplate	Boolean configuration which determines whether or not frames received in the dynamic segment that don't contain a message ID will be admitted into the FIFO.	
M2 Parameter		
Fibex::Fibex4FlexRay::FlexRayFifoConfiguration.admitWithoutMessageId		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
Fr	Fr/FrMultipleConfiguration/FrController/FrFifo	
BSW Parameter		**BSW Type**
FrBaseCycle		EcucIntegerParamDef
BSW Description		
FIFO cycle counter acceptance criteria.		
M2 Template	**M2 Description**	
SystemTemplate	FIFO cycle counter acceptance criteria.	
M2 Parameter		
Fibex::Fibex4FlexRay::FlexRayFifoConfiguration.baseCycle		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
Fr	Fr/FrMultipleConfiguration/FrController/FrFifo	
BSW Parameter		**BSW Type**
FrChannels		EcucEnumerationParamDef
BSW Description		
FIFO channel admittance criteria.		
M2 Template	**M2 Description**	
System Template	The connection between the referencing ECU and the referenced channel via the referenced controller.	
M2 Parameter		
Described by the relation between CommunicationConnector and PhysicalChannel: FibexCore::CoreTopology::CommunicationConnector		
Mapping Rule		**Mapping Type**
If Channel A is referenced set Parameter to FR_CHANNEL_A. If ChannelB is referenced set parameter to FR_CHANNEL_B.		full

BSW Module	BSW Context	
Fr	Fr/FrMultipleConfiguration/FrController/FrFifo	
BSW Parameter		**BSW Type**
FrCycleRepetition		EcucIntegerParamDef
BSW Description		
FIFO cylce counter acceptance criteria. Valid values are 1,2,4,5,8,10,16,20,32,40,50,64. Remark: Values 1,2,4,8,16,32,64 are valid only for FlexRay Protocol 2.1 Rev A compliance.		

M2 Template	M2 Description	
SystemTemplate	FIFO cycle counter acceptance criteria.	
M2 Parameter		
Fibex::Fibex4FlexRay::FlexRayFifoConfiguration.cycleRepetition		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context		
Fr	Fr/FrMultipleConfiguration/FrController/FrFifo		
BSW Parameter		**BSW Type**	
FrFifoDepth		EcucIntegerParamDef	
BSW Description			
Fifo Depth.			
M2 Template	**M2 Description**		
SystemTemplate	Fifo Depth.		
M2 Parameter			
Fibex::Fibex4FlexRay::FlexRayFifoConfiguration.fifoDepth			
Mapping Rule			**Mapping Type**
1:1 mapping			full

BSW Module	BSW Context		
Fr	Fr/FrMultipleConfiguration/FrController/FrFifo		
BSW Parameter		**BSW Type**	
FrMsgIdMask		EcucIntegerParamDef	
BSW Description			
FIFO message identifier acceptance criteria (Mask filter).			
M2 Template	**M2 Description**		
SystemTemplate	FIFO message identifier acceptance criteria (Mask filter).		
M2 Parameter			
Fibex::Fibex4FlexRay::FlexRayFifoConfiguration.msgIdMask			
Mapping Rule			**Mapping Type**
1:1 mapping			full

BSW Module	BSW Context		
Fr	Fr/FrMultipleConfiguration/FrController/FrFifo		
BSW Parameter		**BSW Type**	
FrMsgIdMatch		EcucIntegerParamDef	
BSW Description			
FIFO message identifier acceptance criteria (Match filter).			
M2 Template	**M2 Description**		
SystemTemplate	FIFO message identifier acceptance criteria (Match filter).		
M2 Parameter			
Fibex::Fibex4FlexRay::FlexRayFifoConfiguration.msgIdMatch			
Mapping Rule			**Mapping Type**
1:1 mapping			full

BSW Module	BSW Context	
Fr	Fr/FrMultipleConfiguration/FrController/FrFifo	
BSW Parameter		**BSW Type**

FrRange		EcucParamConfContainerDef	
BSW Description			
FIFO Frame Id range acceptance criteria.			
M2 Template	**M2 Description**		
SystemTemplate	FIFO Frame Id range acceptance criteria.		
M2 Parameter			
Fibex::Fibex4FlexRay::FlexRayFifoRange			
Mapping Rule			**Mapping Type**
create container for each Fifo configuration			full

BSW Module	**BSW Context**		
Fr	Fr/FrMultipleConfiguration/FrController/FrFifo/FrRange		
BSW Parameter		**BSW Type**	
FrRangeMax		EcucIntegerParamDef	
BSW Description			
Last FrameId of this range that will be accepted by the FIFO.			
M2 Template	**M2 Description**		
SystemTemplate	Max Range.		
M2 Parameter			
Fibex::Fibex4FlexRay::FlexRayFifoRange.rangeMax			
Mapping Rule			**Mapping Type**
1:1 mapping			full

BSW Module	**BSW Context**		
Fr	Fr/FrMultipleConfiguration/FrController/FrFifo/FrRange		
BSW Parameter		**BSW Type**	
FrRangeMin		EcucIntegerParamDef	
BSW Description			
First FrameId of this range that will be accepted by the FIFO.			
M2 Template	**M2 Description**		
SystemTemplate	Min Range.		
M2 Parameter			
Fibex::Fibex4FlexRay::FlexRayFifoRange.rangeMin			
Mapping Rule			**Mapping Type**
1:1 mapping			full

BSW Module	**BSW Context**	
Fr	Fr/FrMultipleConfiguration/FrController	
BSW Parameter		**BSW Type**
FrPAllowHaltDueToClock		EcucBooleanParamDef
BSW Description		
Boolean flag that controls the transition to the POC:halt state due to a clock synchronization errors. If set to true, the CC is allowed to transition to POC:halt. If set to false, the CC will not transition to the POC:halt state but will enter or remain in the POC:normal passive state (self healing would still be possible)		
M2 Template	**M2 Description**	
System Template	Boolean flag that controls the transition to the POC:halt state due to a clock synchronization errors. If set to true, the Communication Controller is allowed to transition to POC:halt. If set to false, the Communication Controller will not transition	

M2 Parameter	
Fibex4Flexray::FlexrayTopology::FlexrayCommunicationController.allowHaltDueToClock	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	BSW Context	
Fr	Fr/FrMultipleConfiguration/FrController	
BSW Parameter		**BSW Type**
FrPAllowPassiveToActive		EcucIntegerParamDef
BSW Description		
Number of consecutive even/odd cycle pairs that must have valid clock correction terms before the CC will be allowed to transition from the POC:normal passive state to POC:normal active state. If set to zero, the CC is not allowed to transition from POC:normal passive to POC:normal active		
M2 Template	**M2 Description**	
System Template	Number of consecutive even/odd cycle pairs that must have valid clock correction terms before the Communication Controller will be allowed to transition from the POC:normal passive state to POC:normal active state.	
M2 Parameter		
Fibex4Flexray::FlexrayTopology::FlexrayCommunicationController.allowPassiveToActive		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
Fr	Fr/FrMultipleConfiguration/FrController	
BSW Parameter		**BSW Type**
FrPChannels		EcucEnumerationParamDef
BSW Description		
Channels to which the node is connected. Implementation Type: Fr_ChannelType		
M2 Template	**M2 Description**	
System Template	The connection between the referencing ECU and the referenced channel via the referenced controller.	
M2 Parameter		
Described by the relation between CommunicationConnector and PhysicalChannel: FibexCore::CoreTopology::CommunicationConnector		
Mapping Rule		**Mapping Type**
If Channel A is referenced set Parameter to FR_CHANNEL_A. If ChannelB is referenced set parameter to FR_CHANNEL_B.		full

BSW Module	BSW Context	
Fr	Fr/FrMultipleConfiguration/FrController	
BSW Parameter		**BSW Type**
FrPClusterDriftDamping		EcucIntegerParamDef
BSW Description		
Local cluster drift damping factor used for rate correction [Microticks]. Remark: Upper limit 10 for FlexRay Protocol 3.0 compliance.		
M2 Template	**M2 Description**	
System Template	The cluster drift damping factor used in clock synchronization rate correction in microticks	
M2 Parameter		

Fibex4FlexRay::FlexRayCommunicationController.clusterDriftDamping	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	**BSW Context**
Fr	Fr/FrMultipleConfiguration/FrController
BSW Parameter	**BSW Type**
FrPDecodingCorrection	EcucIntegerParamDef
BSW Description	
Value used by the receiver to calculate the difference between primary time reference point and secondary time reference point [Microticks]. Remark: Lower limit 14 for FlexRay Protocol 2.1 Rev. A compliance. Upper limit 136 for FlexRay Protocol 3.0 compliance.	
M2 Template	**M2 Description**
System Template	Value used by the receiver to calculate the difference between primary time reference point and secondary time reference point. Unit: Microticks (pDecodingCorrection)
M2 Parameter	
Fibex4Flexray::FlexrayTopology::FlexrayCommunicationController.decodingCorrection	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	**BSW Context**
Fr	Fr/FrMultipleConfiguration/FrController
BSW Parameter	**BSW Type**
FrPDelayCompensationA	EcucIntegerParamDef
BSW Description	
Value used to compensate for reception delays on the indicated channel. This covers assumed propagation delay up to cPropagationDelayMax for microticks in the range of 0.0125us to 0.05us [Microticks]. Remark: Lower limit 4 for FlexRay Protocol 3.0 compliance. Remark: Upper limit 200 for FlexRay Protocol 2.1 Rev A compliance.	
M2 Template	**M2 Description**
System Template	Value used to compensate for reception delays on channel A Unit: Microticks
M2 Parameter	
Fibex4Flexray::FlexrayTopology::FlexrayCommunicationController.delayCompensationA	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	**BSW Context**
Fr	Fr/FrMultipleConfiguration/FrController
BSW Parameter	**BSW Type**
FrPDelayCompensationB	EcucIntegerParamDef
BSW Description	
Value used to compensate for reception delays on the indicated channel. This covers assumed propagation delay up to cPropagationDelayMax for microticks in the range of 0.0125us to 0.05us [Microticks]. Remark: Lower limit 4 for FlexRay Protocol 3.0 compliance. Remark: Upper limit 200 for FlexRay Protocol 2.1 Rev A compliance.	
M2 Template	**M2 Description**
System Template	Value used to compensate for reception delays on channel B. Unit: Microticks

M2 Parameter	
Fibex4Flexray::FlexrayTopology::FlexrayCommunicationController.delayCompensationB	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	BSW Context
Fr	Fr/FrMultipleConfiguration/FrController
BSW Parameter	**BSW Type**
FrPExternalSync	EcucBooleanParamDef
BSW Description	
Flag indicating whether the node is externally synchronized (operating as time gateway sink in an TT-E cluster) or locally synchronized. If FrPExternalSync is set to 'true' then FrPTwoKeySlotMode must also be set to 'true'. Remarks: Set to 'false' for FlexRay Protocol 2.1 Rev. A compliance.	
M2 Template	**M2 Description**
SystemTemplate	Flag indicating whether the node is externally synchronized (operating as Time Gateway Sink in an TT-E Time Triggered External Sync cluster) or locally synchronized.
M2 Parameter	
Fibex4Flexray::FlexrayTopology::FlexrayCommunicationController.externalSync	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	BSW Context
Fr	Fr/FrMultipleConfiguration/FrController
BSW Parameter	**BSW Type**
FrPFallBackInternal	EcucBooleanParamDef
BSW Description	
Flag indicating whether a time gateway sink node will switch to local clock operation when synchronization with the time gateway source node is lost (FrPFallBackInternal = true) or will instead go to POC:ready (FrPFallBackInternal =false). Remarks: Set to 'false' for FlexRay Protocol 2.1 Rev. A compliance.	
M2 Template	**M2 Description**
SystemTemplate	Flag indicating whether a Time Gateway Sink node will switch to local clock operation when synchronization with the Time Gateway Source node is lost (pFallBackInternal = true) or will instead go to POC:ready (pFallBackInternal = false).
M2 Parameter	
Fibex4Flexray::FlexrayTopology::FlexrayCommunicationController.fallBackInternal	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	BSW Context
Fr	Fr/FrMultipleConfiguration/FrController
BSW Parameter	**BSW Type**
FrPKeySlotId	EcucIntegerParamDef
BSW Description	
ID of the key slot, i.e., the slot used to transmit the startup frame, sync frame, or designated key slot frame. If this parameter is set to zero the node does not have a key slot.	
M2 Template	**M2 Description**

System Template	ID of the slot used to transmit the startup frame, sync frame, or designated single slot frame.
M2 Parameter	
Fibex4Flexray::FlexrayTopology::FlexrayCommunicationController.keySlotID	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	**BSW Context**
Fr	Fr/FrMultipleConfiguration/FrController
BSW Parameter	**BSW Type**
FrPKeySlotOnlyEnabled	EcucBooleanParamDef
BSW Description	
Flag indicating whether or not the node shall enter key slot only mode following startup. Remarks: This parameter maps to FlexRay Protocol 2.1 Rev. A parameter pSingleSlotEnabled.	
M2 Template	**M2 Description**
System Template	Flag indicating whether or not the node shall enter single slot mode following startup.
M2 Parameter	
Fibex4Flexray::FlexrayTopology::FlexrayCommunicationController.keySlotOnlyEnabled	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	**BSW Context**
Fr	Fr/FrMultipleConfiguration/FrController
BSW Parameter	**BSW Type**
FrPKeySlotUsedForStartup	EcucBooleanParamDef
BSW Description	
Flag indicating whether the key slot is used to transmit a startup frame. If FrPKeySlotUsedForStartup is set to true then FrPKeySlotUsedForSync must also be set to true. If FrPTwoKeySlotMode is set to true then both FrPKeySlotUsedForSync and FrPKeySlotUsedForStartup must also be set to true.	
M2 Template	**M2 Description**
System Template	Flag indicating whether the Key Slot is used to transmit a startup frame.
M2 Parameter	
Fibex4Flexray::FlexrayTopology::FlexrayCommunicationController.keySlotUsedForStartUp	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	**BSW Context**
Fr	Fr/FrMultipleConfiguration/FrController
BSW Parameter	**BSW Type**
FrPKeySlotUsedForSync	EcucBooleanParamDef
BSW Description	
Flag indicating whether the key slot is used to transmit a sync frame. If FrPKeySlotUsedForStartup is set to true then FrPKeySlotUsedForSync must also be set to true. If FrPTwoKeySlotMode is set to true then both FrPKeySlotUsedForSync and FrPKeySlotUsedForStartup must also be set to true.	
M2 Template	**M2 Description**
System Template	Flag indicating whether the Key Slot is used to transmit a sync frame.
M2 Parameter	

Fibex4Flexray::FlexrayTopology::FlexrayCommunicationController.keySlotUsed-ForSync	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	**BSW Context**	
Fr	Fr/FrMultipleConfiguration/FrController	
BSW Parameter		**BSW Type**
FrPLatestTx		EcucIntegerParamDef
BSW Description		
Number of the last minislot in which a frame transmission can start in the dynamic segment. Remark: Upper limit 7980 for FlexRay Protocol 2.1 Rev A compliance.		
M2 Template	**M2 Description**	
System Template	The number of the last minislot in which a transmission can start in the dynamic segment for the respective node	
M2 Parameter		
Fibex4Flexray::FlexrayTopology::FlexrayCommunicationController.latestTX		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
Fr	Fr/FrMultipleConfiguration/FrController	
BSW Parameter		**BSW Type**
FrPMacroInitialOffsetA		EcucIntegerParamDef
BSW Description		
Integer number of macroticks between the static slot boundary and the following macrotick boundary of the secondary time reference point based on the nominal macrotick duration [Macroticks].		
M2 Template	**M2 Description**	
System Template	Integer number of macroticks between the static slot boundary and the closest macrotick boundary of the secondary time reference point based on the nominal macrotick duration. (pMacroInitialOffset)	
M2 Parameter		
Fibex4Flexray::FlexrayTopology::FlexrayCommunicationController.macroInitialOffsetA		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
Fr	Fr/FrMultipleConfiguration/FrController	
BSW Parameter		**BSW Type**
FrPMacroInitialOffsetB		EcucIntegerParamDef
BSW Description		
Integer number of macroticks between the static slot boundary and the following macrotick boundary of the secondary time reference point based on the nominal macrotick duration [Macroticks].		
M2 Template	**M2 Description**	
System Template	Integer number of macroticks between the static slot boundary and the closest macrotick boundary of the secondary time reference point based on the nominal macrotick duration. (pMacroInitialOffset)	
M2 Parameter		
Fibex4Flexray::FlexrayTopology::FlexrayCommunicationController.macroInitialOffsetB		
Mapping Rule		**Mapping Type**
1:1 mapping		full

— AUTOSAR CONFIDENTIAL —

BSW Module	BSW Context		
Fr	Fr/FrMultipleConfiguration/FrController		
BSW Parameter		**BSW Type**	
FrPMicroInitialOffsetA		EcucIntegerParamDef	
BSW Description			
Number of microticks between the secondary time reference point and the macrotick boundary immediately following the secondary time reference point. The parameter depends on FrPDelayCompensationA and therefore it has to be set independently for each channel [Microticks].			
M2 Template	**M2 Description**		
System Template	Number of microticks between the closest macrotick boundary described by gMacroInitialOffset and the secondary time reference point. The parameter depends on pDelayCompensationA and therefore it has to be set independently for each channel.		
M2 Parameter			
Fibex4Flexray::FlexrayTopology::FlexrayCommunicationController.microInitialOffsetA			
Mapping Rule			**Mapping Type**
1:1 mapping			full

BSW Module	BSW Context		
Fr	Fr/FrMultipleConfiguration/FrController		
BSW Parameter		**BSW Type**	
FrPMicroInitialOffsetB		EcucIntegerParamDef	
BSW Description			
Number of microticks between the secondary time reference point and the macrotick boundary immediately following the secondary time reference point. The parameter depends on FrPDelayCompensationB and therefore it has to be set independently for each channel [Microticks].			
M2 Template	**M2 Description**		
System Template	Number of microticks between the closest macrotick boundary described by gMacroInitialOffset and the secondary time reference point. The parameter depends on pDelayCompensationB and therefore it has to be set independently for each channel.		
M2 Parameter			
Fibex4Flexray::FlexrayTopology::FlexrayCommunicationController.microInitialOffsetB			
Mapping Rule			**Mapping Type**
1:1 mapping			full

BSW Module	BSW Context	
Fr	Fr/FrMultipleConfiguration/FrController	
BSW Parameter		**BSW Type**
FrPMicroPerCycle		EcucIntegerParamDef
BSW Description		
Nominal number of microticks in the communication cycle of the local node. If nodes have different microtick durations this number will differ from node to node [Microticks]. Remark: Lower limit 960 for FlexRay Protocol 3.0 compliance. Upper limit 640000 for FlexRay Protocol 2.1 Rev A compliance.		
M2 Template	**M2 Description**	
System Template	The nominal number of microticks in a communication cycle	
M2 Parameter		
Fibex4Flexray::FlexrayTopology::FlexrayCommunicationController.microPerCycle		

Mapping Rule	Mapping Type
1:1 mapping	full

BSW Module	BSW Context	
Fr	Fr/FrMultipleConfiguration/FrController	
BSW Parameter		**BSW Type**
FrPNmVectorEarlyUpdate		EcucBooleanParamDef
BSW Description		
Flag indicating when the update of the Network Management Vector in the CHI shall take place. If FrPNmVectorEarlyUpdate is set to false, the update shall take place after the NIT. If FrPNmVectorEarlyUpdate is set to true, the update shall take place after the end of the static segment. Remarks: Set to 'false' for FlexRay Protocol 2.1 Rev. A compliance.		
M2 Template	**M2 Description**	
SystemTemplate	Flag indicating when the update of the Network Management Vector in the CHI shall take place.	
M2 Parameter		
Fibex4Flexray::FlexrayTopology::FlexrayCommunicationController.nmVectorEarlyUpdate		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
Fr	Fr/FrMultipleConfiguration/FrController	
BSW Parameter		**BSW Type**
FrPOffsetCorrectionOut		EcucIntegerParamDef
BSW Description		
Magnitude of the maximum permissible offset correction value [Microticks]. Remark: Upper limit 15567 for FlexRay Protocol 2.1 Rev A compliance. Remark: Lower limit 15 for FlexRay Protocol 3.0 compliance.		
M2 Template	**M2 Description**	
System Template	Magnitude of the maximum permissible offset correction value. Unit:microtick (pOffsetCorrectionOut)	
M2 Parameter		
Fibex4Flexray::FlexrayTopology::FlexrayCommunicationController.offsetCorrectionOut		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
Fr	Fr/FrMultipleConfiguration/FrController	
BSW Parameter		**BSW Type**
FrPOffsetCorrectionStart		EcucIntegerParamDef
BSW Description		
Start of the offset correction phase within the NIT, expressed as the number of macroticks from the start of cycle [Macroticks]. Remark: This parameter maps to FlexRay Protocol 2.1 Rev. A parameter gOffsetCorrectionStart. Remark: Lower limit 9 for FlexRay Protocol 2.1 Rev A compliance.		
M2 Template	**M2 Description**	
SystemTemplate	Start of the offset correction phase within the NIT, expressed as the number of macroticks from the start of cycle	
M2 Parameter		
Fibex4Flexray::FlexrayTopology::FlexrayCommunicationCluster.offsetCorrectionStart		

Mapping Rule	Mapping Type
1:1 mapping	full

BSW Module	BSW Context	
Fr	Fr/FrMultipleConfiguration/FrController	
BSW Parameter		**BSW Type**
FrPPayloadLengthDynMax		EcucIntegerParamDef
BSW Description		
Maximum payload length for dynamic frames [16 bit words].		
M2 Template	**M2 Description**	
System Template	Maximum payload length for the dynamic channel of a frame in 16 bit WORDS.	
M2 Parameter		
Fibex4Flexray::FlexrayTopology::FlexrayCommunicationController.maximumDynamicPayloadLength		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
Fr	Fr/FrMultipleConfiguration/FrController	
BSW Parameter		**BSW Type**
FrPRateCorrectionOut		EcucIntegerParamDef
BSW Description		
Magnitude of the maximum permissible rate correction value and the maximum drift offset between two nodes operating with unsynchronized clocks for one communication cycle [Microticks]. Remarks: This parameter maps to FlexRay Protocol 2.1 Rev. A parameter pdMaxDrift. Lower limit 3 for FlexRay Protocol 3.0 compliance. Upper limit 1923 for FlexRay Protocol 2.1 Rev A compliance.		
M2 Template	**M2 Description**	
System Template	Magnitude of the maximum permissible rate correction value and the maximum drift offset between two nodes operating with unsynchronized clocks for one communication cycle. Unit:Microticks (pRateCorrectionOut)	
M2 Parameter		
Fibex4Flexray::FlexrayTopology::FlexrayCommunicationController.rateCorrectionOut		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
Fr	Fr/FrMultipleConfiguration/FrController	
BSW Parameter		**BSW Type**
FrPSamplesPerMicrotick		EcucEnumerationParamDef
BSW Description		
Number of samples per microtick. Remark: Allowed range N1SAMPLES, N2SAMPLES for FlexRay Protocol 3.0 compliance.		
M2 Template	**M2 Description**	
System Template	Number of samples per microtick	
M2 Parameter		
Fibex4Flexray::FlexrayTopology::FlexrayCommunicationController.samplesPerMicrotick		
Mapping Rule		**Mapping Type**
1:1 mapping		full

— AUTOSAR CONFIDENTIAL —

BSW Module	BSW Context	
Fr	Fr/FrMultipleConfiguration/FrController	
BSW Parameter		**BSW Type**
FrPSecondKeySlotId		EcucIntegerParamDef
BSW Description		
ID of the second key slot, in which a second startup frame shall be sent when operating as a coldstart node in a TT-L or TT-D cluster. If this parameter is set to zero the node does not have a second key slot. Remark: Set to 0 for FlexRay Protocol 2.1 Rev A compliance.		
M2 Template	**M2 Description**	
SystemTemplate	ID of the second Key slot, in which a second startup frame shall be sent in TT-L Time Triggered Local Master Sync or TT-E Time Triggered External Sync mode. If this parameter is set to zero the node does not have a second key slot.	
M2 Parameter		
Fibex4Flexray::FlexrayTopology::FlexrayCommunicationController.secondkeySlotID		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
Fr	Fr/FrMultipleConfiguration/FrController	
BSW Parameter		**BSW Type**
FrPTwoKeySlotMode		EcucBooleanParamDef
BSW Description		
Flag indicating whether node operates as a coldstart node in a TT-E or TT-L cluster. If pTwoKeySlotMode is set to true then both pKeySlotUsedForSync and pKeySlotUsedForStartup must also be set to true. If pExternalSync is set to true then pTwoKeySlotMode must also be set to true. Remark: Set to false for FlexRay Protocol 2.1 Rev A compliance.		
M2 Template	**M2 Description**	
SystemTemplate	Flag indicating whether node operates as a startup node in a TT-E Time Triggered External Sync or TT-L Time Triggered Local Master Sync cluster.	
M2 Parameter		
Fibex4Flexray::FlexrayTopology::FlexrayCommunicationController.twoKeySlotMode		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
Fr	Fr/FrMultipleConfiguration/FrController	
BSW Parameter		**BSW Type**
FrPWakeupChannel		EcucEnumerationParamDef
BSW Description		
Channel used by the node to send a wakeup pattern. FrPWakeupChannel must be selected from among the channels configured by FrPChannels.		
M2 Template	**M2 Description**	
System Template	Referenced channel used by the node to send a wakeup pattern. (pWakeupChannel) / True: Channel A; False: Channel B	
M2 Parameter		
Fibex4FlexRay::FlexRayTopology::FlexRayCommunicationConnector.wakeUpChannel		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context		
Fr	Fr/FrMultipleConfiguration/FrController		
BSW Parameter		**BSW Type**	
FrPWakeupPattern		EcucIntegerParamDef	
BSW Description			
Number of repetitions of the wakeup symbol that are combined to form a wakeup pattern when the node enters the POC:wakeup send state. Remark: Lower limit 2 for FlexRay Protocol 2.1 Rev A compliance.			
M2 Template	**M2 Description**		
System Template	Number of repetitions of the Tx-wakeup symbol to be sent during the CC_WakeupSend state of this Node in the cluster		
M2 Parameter			
Fibex4Flexray::FlexrayTopology::FlexrayCommunicationController.wakeUpPattern			
Mapping Rule			**Mapping Type**
1:1 mapping			full

BSW Module	BSW Context		
Fr	Fr/FrMultipleConfiguration/FrController		
BSW Parameter		**BSW Type**	
FrPdAcceptedStartupRange		EcucIntegerParamDef	
BSW Description			
Expanded range of measured clock deviation allowed for startup frames during integration [Microticks]. Remark: Upper limit 1875 for FlexRay Protocol 2.1 Rev A compliance. Remark: Lower limit 29 for FlexRay Protocol 3.0 compliance.			
M2 Template	**M2 Description**		
System Template	Expanded range of measured clock deviation allowed for startup frames during integration. Unit:microtick		
M2 Parameter			
Fibex4Flexray::FlexrayTopology::FlexrayCommunicationController.acceptedStartupRange			
Mapping Rule			**Mapping Type**
1:1 mapping			full

BSW Module	BSW Context		
Fr	Fr/FrMultipleConfiguration/FrController		
BSW Parameter		**BSW Type**	
FrPdListenTimeout		EcucIntegerParamDef	
BSW Description			
Value for the startup listen timeout and wakeup listen timeout. Although this is a node local parameter, the real time equivalent of this value should be the same for all nodes in the cluster [Microticks]. Remark: Lower limit 1926 for FlexRay Protocol 3.0 compliance. Upper limit 1283846 for FlexRay Protocol 2.1 Rev. A compliance.			
M2 Template	**M2 Description**		
System Template	Upper limit for the startup listen timeout and wakeup listen timeout. Unit: Microticks		
M2 Parameter			
Fibex4Flexray::FlexrayTopology::FlexrayCommunicationController.listenTimeout			
Mapping Rule			**Mapping Type**
1:1 mapping			full

BSW Module	BSW Context

— AUTOSAR CONFIDENTIAL —

Fr	Fr/FrMultipleConfiguration/FrController
BSW Parameter	**BSW Type**
FrPdMicrotick	EcucEnumerationParamDef
BSW Description	
Duration of a microtick. Remark: Allowed range T12_5NS, T25NS, T50NS for FlexRay Protocol 3.0 compliance.	
M2 Template	**M2 Description**
System Template	Duration of a microtick. This attribute can be derived from samplePerMicrotick and gdSampleClockPeriod. Unit: seconds
M2 Parameter	
Fibex4Flexray::FlexrayTopology::FlexrayCommunicationController.microtickDuration	
Mapping Rule	**Mapping Type**
1:1 mapping	full

D.7 FlexRay Interface Mapping

BSW Module	**BSW Context**
FrIf	FrIf
BSW Parameter	**BSW Type**
FrIfConfig	EcucParamConfContainerDef
BSW Description	
Configuration of the FlexRay Interface. This container is a MultipleConfigurationContainer, i.e. this container and its sub-containers exist once per configuration set.	
M2 Template	**M2 Description**
System Template	The CommunicationCluster is the main element to describe the topological connection
M2 Parameter	
SystemTemplate:Fibex:Fibex4FlexRay:FlexRayCluster	
Mapping Rule	**Mapping Type**
Container must be created if the ECU is connected to a FlexRay Cluster	full

BSW Module	**BSW Context**
FrIf	FrIf/FrIfConfig
BSW Parameter	**BSW Type**
FrIfCluster	EcucParamConfContainerDef
BSW Description	
This container specifies a FrIf Cluster and all related data which is required to enable communication of the Cluster. A Cluster may consist of more than one Controller.	
M2 Template	**M2 Description**
System Template	The CommunicationCluster is the main element to describe the topological connection
M2 Parameter	
Fibex::Fibex4FlexRay::FlexRayCluster	
Mapping Rule	**Mapping Type**
Container must be created if the ECU is connected to a FlexRay Cluster	full

BSW Module	**BSW Context**
FrIf	FrIf/FrIfConfig/FrIfCluster
BSW Parameter	**BSW Type**
FrIfClstIdx	EcucIntegerParamDef

BSW Description		
This parameter provides a zero-based consecutive index of the FlexRay Clusters. Upper layer BSW modules and the FrIf itself use this index to identify a FlexRay Cluster.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
FrIf	FrIf/FrIfConfig/FrIfCluster	
BSW Parameter		**BSW Type**
FrIfClusterDemEventParameterRefs		EcucParamConfContainerDef
BSW Description		
Container for the references to DemEventParameter elements which shall be invoked using the API Dem_ReportErrorStatus API in case the corresponding error occurs. The EventId is taken from the referenced DemEventParameter's DemEventId value. The standardized errors are provided in the container and can be extended by vendor specific error references.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
FrIf	FrIf/FrIfConfig/FrIfCluster/FrIfClusterDemEventParameterRefs	
BSW Parameter		**BSW Type**
FRIF_E_ACS_CH_A		EcucSymbolicNameReferenceDef
BSW Description		
Reference to the DemEventParameter which shall be issued when an error in ACS on channel A was detected. If the reference is not configured the error shall not be reported (neither to DET nor to DEM).		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
FrIf	FrIf/FrIfConfig/FrIfCluster/FrIfClusterDemEventParameterRefs	
BSW Parameter		**BSW Type**
FRIF_E_ACS_CH_B		EcucSymbolicNameReferenceDef
BSW Description		
Reference to the DemEventParameter which shall be issued when an error in ACS on channel B was detected. If the reference is not configured the error shall not be reported (neither to DET nor to DEM).		
M2 Template	**M2 Description**	

M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**
FrIf	FrIf/FrIfConfig/FrIfCluster/FrIfClusterDemEventParameterRefs
BSW Parameter	**BSW Type**
FRIF_E_NIT_CH_A	EcucSymbolicNameReferenceDef
BSW Description	
Reference to the DemEventParameter which shall be issued when an error in NIT on channel A was detected. If the reference is not configured the error shall not be reported (neither to DET nor to DEM).	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**
FrIf	FrIf/FrIfConfig/FrIfCluster/FrIfClusterDemEventParameterRefs
BSW Parameter	**BSW Type**
FRIF_E_NIT_CH_B	EcucSymbolicNameReferenceDef
BSW Description	
Reference to the DemEventParameter which shall be issued when an error in NIT on channel B was detected. If the reference is not configured the error shall not be reported (neither to DET nor to DEM).	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**
FrIf	FrIf/FrIfConfig/FrIfCluster/FrIfClusterDemEventParameterRefs
BSW Parameter	**BSW Type**
FRIF_E_SW_CH_A	EcucSymbolicNameReferenceDef
BSW Description	
Reference to the DemEventParameter which shall be issued when an error in SW on channel A was detected. If the reference is not configured the error shall not be reported (neither to DET nor to DEM).	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	BSW Context	
FrIf	FrIf/FrIfConfig/FrIfCluster/FrIfClusterDemEventParameterRefs	
BSW Parameter		**BSW Type**
FRIF_E_SW_CH_B		EcucSymbolicNameReferenceDef
BSW Description		
Reference to the DemEventParameter which shall be issued when an error in SW on channel B was detected. If the reference is not configured the error shall not be reported (neither to DET nor to DEM).		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
FrIf	FrIf/FrIfConfig/FrIfCluster	
BSW Parameter		**BSW Type**
FrIfController		EcucParamConfContainerDef
BSW Description		
This container contains the configuration of FlexRay CC.		
M2 Template	**M2 Description**	
System Template	The communication controller is a dedicated hardware device by means of which hosts are sending frames to and receiving frames from the communication medium.	
M2 Parameter		
Fibex:FibexCore::Topology::EcuInstance::CommunicationController		
Mapping Rule		**Mapping Type**
Container must be created if the ECU contains a FlexRay communication controller that is connected to the regarded communication cluster.		full

BSW Module	BSW Context	
FrIf	FrIf/FrIfConfig/FrIfCluster/FrIfController	
BSW Parameter		**BSW Type**
FrIfCtrlIdx		EcucIntegerParamDef
BSW Description		
This parameter provides a zero-based consecutive index of the FlexRay Communication Controllers. Upper layer BSW modules and the FrIf itself use this index to identify a FlexRay CC.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
FrIf	FrIf/FrIfConfig/FrIfCluster/FrIfController	
BSW Parameter		**BSW Type**
FrIfFrCtrlRef		EcucSymbolicNameReferenceDef
BSW Description		

Reference to a Controller, which is handled by a specific Driver. This reference is unique for the ECU.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
FrIf	FrIf/FrIfConfig/FrIfCluster/FrIfController	
BSW Parameter		**BSW Type**
FrIfFrameTriggering		EcucParamConfContainerDef
BSW Description		
A Frame triggering contains the communication parameters of the FlexRay Frame as well as a reference to the Frame Construction Plan.		
M2 Template	**M2 Description**	
System Template	The FrameTriggering describes the instance of a frame sent on a channel and defines the manner of triggering (timing information) and identification of a frame on the channel, on which it is sent.	
M2 Parameter		
Fibex4FlexRay::FlexRayFrameTriggering		
Mapping Rule		**Mapping Type**
Container must be created for each existing FlexRayFrameTriggering element that is connected to a CommConnectorPort of the regarded communication controller.		full

BSW Module	**BSW Context**	
FrIf	FrIf/FrIfConfig/FrIfCluster/FrIfController/FrIfFrameTriggering	
BSW Parameter		**BSW Type**
FrIfAllowDynamicLSduLength		EcucBooleanParamDef
BSW Description		
Allows L-PDU length reduction ('FrIfLSduLength' defines max. length) and indicates that the related CC buffer has to be reconfigured for the actual length and Header-CRC before transmission of the L-PDU.		
M2 Template	**M2 Description**	
SystemTemplate	Allows L-PDU length reduction and indicates that the related CC buffer has to be reconfigured for the actual length and Header-CRC before transmission of the L-PDU.	
M2 Parameter		
Fibex4FlexRay::Communication::FlexRayFrameTriggering.allowDynamicLSduLength		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
FrIf	FrIf/FrIfConfig/FrIfCluster/FrIfController/FrIfFrameTriggering	
BSW Parameter		**BSW Type**
FrIfAlwaysTransmit		EcucBooleanParamDef
BSW Description		
Defines wether the driver's API function Fr_TransmitTxLPdu() shall always be called for this L-PDU.		
M2 Template	**M2 Description**	

M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**	
FrIf	FrIf/FrIfConfig/FrIfCluster/FrIfController/FrIfFrameTriggering	
BSW Parameter		**BSW Type**
FrIfBaseCycle		EcucIntegerParamDef
BSW Description		
This parameter contains the FlexRay Base Cycle used to transmit this FlexRay Frame.		
M2 Template	**M2 Description**	
System Template	The first communication cycle where the frame is sent. This value is incremented at the beginning of each new cycle, ranging from 0 to 63, and is reset to 0 after a sequence of 64 cycles.	
M2 Parameter		
Fibex4FlexRay::FlexRayFrameTriggering::FlexrayAbsolutelyScheduledTiming::CycleRepetition.baseCycle		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
FrIf	FrIf/FrIfConfig/FrIfCluster/FrIfController/FrIfFrameTriggering	
BSW Parameter		**BSW Type**
FrIfChannel		EcucEnumerationParamDef
BSW Description		
This parameter contains the FlexRay Channel used to transmit this FlexRay Frame.		
M2 Template	**M2 Description**	
System Template	One frame triggering is defined for exactly one channel. Channels may have assigned an arbitrary number of frame triggerings.	
M2 Parameter		
PhysicalChannel.frameTriggering		
Mapping Rule		**Mapping Type**
FrameTriggering element in the System Template is aggregated by the Physical Channel that is used to transmit this FlexRay Frame		full

BSW Module	**BSW Context**	
FrIf	FrIf/FrIfConfig/FrIfCluster/FrIfController/FrIfFrameTriggering	
BSW Parameter		**BSW Type**
FrIfCycleRepetition		EcucIntegerParamDef
BSW Description		
This parameter contains the FlexRay Cycle Repetition used to transmit this FlexRay Frame.. possible Values: 1,2,4,8,16,32,64		
M2 Template	**M2 Description**	
System Template	The number of communication cycles (after the first cycle) whenever the frame described by this timing is sent again.	
M2 Parameter		
Fibex4FlexRay::FlexRayFrameTriggering::FlexrayAbsolutelyScheduledTiming::CycleRepetition.cycleRepetition		

Mapping Rule	Mapping Type
1:1 mapping	full

BSW Module	BSW Context		
FrIf	FrIf/FrIfConfig/FrIfCluster/FrIfController/FrIfFrameTriggering		
BSW Parameter		**BSW Type**	
FrIfFrameStructureRef		EcucReferenceDef	
BSW Description			
Reference to the Construction Plan of the FlexRay Frame.			
M2 Template	**M2 Description**		
System Template	One frame can be triggered on different channels. If a frame has no frame triggering, it won't be sent at all. A frame triggering has assigned exactly one frame, which it triggers.		
M2 Parameter			
CoreCommunication::PhysicalChannel::FrameTriggering.frame			
Mapping Rule			**Mapping Type**
Reference must comply to the reference in the System Description between the FrameTriggering element and the Frame.element			full

BSW Module	BSW Context		
FrIf	FrIf/FrIfConfig/FrIfCluster/FrIfController/FrIfFrameTriggering		
BSW Parameter		**BSW Type**	
FrIfFrameTriggeringDemEventParameterRefs		EcucParamConfContainerDef	
BSW Description			
Container for the references to DemEventParameter elements which shall be invoked using the API Dem_ReportErrorStatus API in case the corresponding error occurs. The EventId is taken from the referenced DemEventParameter's DemEventId value. The standardized errors are provided in the container and can be extended by vendor specific error references.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	BSW Context		
FrIf	FrIf/FrIfConfig/FrIfCluster/FrIfController/FrIfFrameTriggering/FrIfFrameTriggeringDemEventParameterRefs		
BSW Parameter		**BSW Type**	
FrIfDemFTSlotStatusRef		EcucSymbolicNameReferenceDef	
BSW Description			
Reference to DEM event Id that is reported when FlexRay driver module detects slot errors. If this parameter is not configured, no event reporting happens.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

<table>
<tr><th>BSW Module</th><th colspan="2">BSW Context</th></tr>
<tr><td>FrIf</td><td colspan="2">FrIf/FrIfConfig/FrIfCluster/FrIfController/FrIfFrameTriggering</td></tr>
<tr><th colspan="2">BSW Parameter</th><th>BSW Type</th></tr>
<tr><td colspan="2">FrIfLSduLength</td><td>EcucIntegerParamDef</td></tr>
<tr><th colspan="3">BSW Description</th></tr>
<tr><td colspan="3">The payload length of the Frame is given here. This parameter is required for validation if configured PDUs and update information fits into the Frame at configuration time [bytes].</td></tr>
<tr><th>M2 Template</th><th colspan="2">M2 Description</th></tr>
<tr><td>System Template</td><td colspan="2">The used length (in bytes) of the referencing frame. Should not be confused with a static byte length reserved for each frame by some platforms (e.g. FlexRay).</td></tr>
<tr><th colspan="3">M2 Parameter</th></tr>
<tr><td colspan="3">CoreCommunication::Communication::Frame.frameLength</td></tr>
<tr><th colspan="2">Mapping Rule</th><th>Mapping Type</th></tr>
<tr><td colspan="2">Find Frame that is referenced by the regarded FrameTriggering and use the frameLength attribute</td><td>full</td></tr>
</table>

<table>
<tr><th>BSW Module</th><th colspan="2">BSW Context</th></tr>
<tr><td>FrIf</td><td colspan="2">FrIf/FrIfConfig/FrIfCluster/FrIfController/FrIfFrameTriggering</td></tr>
<tr><th colspan="2">BSW Parameter</th><th>BSW Type</th></tr>
<tr><td colspan="2">FrIfMessageId</td><td>EcucIntegerParamDef</td></tr>
<tr><th colspan="3">BSW Description</th></tr>
<tr><td colspan="3">The first two bytes of the payload segment of the FlexRay frame format for frames transmitted in the dynamic segment can be used as receiver filterable data called the message ID.</td></tr>
<tr><th>M2 Template</th><th colspan="2">M2 Description</th></tr>
<tr><td>SystemTemplate</td><td colspan="2">The first two bytes of the payload segment of the FlexRay frame format for frames transmitted in the</td></tr>
<tr><th colspan="3">M2 Parameter</th></tr>
<tr><td colspan="3">Fibex::Fibex4FlexRay::FlexrayCommunication::FlexrayFrameTriggering.messageId</td></tr>
<tr><th colspan="2">Mapping Rule</th><th>Mapping Type</th></tr>
<tr><td colspan="2">1:1 mapping</td><td>full</td></tr>
</table>

<table>
<tr><th>BSW Module</th><th colspan="2">BSW Context</th></tr>
<tr><td>FrIf</td><td colspan="2">FrIf/FrIfConfig/FrIfCluster/FrIfController/FrIfFrameTriggering</td></tr>
<tr><th colspan="2">BSW Parameter</th><th>BSW Type</th></tr>
<tr><td colspan="2">FrIfPayloadPreamble</td><td>EcucBooleanParamDef</td></tr>
<tr><th colspan="3">BSW Description</th></tr>
<tr><td colspan="3">Switching the Payload Preamble bit.</td></tr>
<tr><th>M2 Template</th><th colspan="2">M2 Description</th></tr>
<tr><td>System Template</td><td colspan="2">Switching the Payload Preamble bit.</td></tr>
<tr><th colspan="3">M2 Parameter</th></tr>
<tr><td colspan="3">Fibex4FlexRay::Communication::FlexRayFrameTriggering.payloadPreambleIndicator</td></tr>
<tr><th colspan="2">Mapping Rule</th><th>Mapping Type</th></tr>
<tr><td colspan="2">1:1 mapping</td><td>full</td></tr>
</table>

<table>
<tr><th>BSW Module</th><th colspan="2">BSW Context</th></tr>
<tr><td>FrIf</td><td colspan="2">FrIf/FrIfConfig/FrIfCluster/FrIfController/FrIfFrameTriggering</td></tr>
<tr><th colspan="2">BSW Parameter</th><th>BSW Type</th></tr>
<tr><td colspan="2">FrIfSlotId</td><td>EcucIntegerParamDef</td></tr>
<tr><th colspan="3">BSW Description</th></tr>
<tr><td colspan="3">This parameter contains the FlexRay Slot ID used to transmit this FlexRay Frame.</td></tr>
</table>

M2 Template	M2 Description	
System Template	In the static part the SlotID defines the slot in which the frame is transmitted. In the dynamic part, the slot id is equivalent to a priority.	
M2 Parameter		
Fibex4FlexRay::FlexRayFrameTriggering::FlexrayAbsolutelyScheduledTiming.slotId		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context		
FrIf	FrIf/FrIfConfig/FrIfCluster/FrIfController		
BSW Parameter		**BSW Type**	
FrIfLPdu		EcucParamConfContainerDef	
BSW Description			
Reference to a L-PDU index			
M2 Template	**M2 Description**		
System Template	Data frame which is sent over a communication medium. This element describes the pure Layout of a frame sent on a channel.		
M2 Parameter			
CoreCommunication::Frame			
Mapping Rule			**Mapping Type**
Create container for each FlexRay Frame that is transmitted or received via the regarded communication controller..			full

BSW Module	BSW Context		
FrIf	FrIf/FrIfConfig/FrIfCluster/FrIfController/FrIfLPdu		
BSW Parameter		**BSW Type**	
FrIfLPduIdx		EcucIntegerParamDef	
BSW Description			
This parameter identifies the L-PDU in the interaction between FlexRay Interface and FlexRay Driver.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	BSW Context		
FrIf	FrIf/FrIfConfig/FrIfCluster/FrIfController/FrIfLPdu		
BSW Parameter		**BSW Type**	
FrIfReconfigurable		EcucBooleanParamDef	
BSW Description			
This parameter specifies that this LPdu is reconfigurable using FrIf_ReconfigLPdu. This means that this LPdu can be assigned to a different FrameTriggering at runtime. However, this reconfiguration is limited by hardware constraints. The direction of the LPdu cannot be reconfigured.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

— AUTOSAR CONFIDENTIAL —

BSW Module	BSW Context	
FrIf	FrIf/FrIfConfig/FrIfCluster/FrIfController/FrIfLPdu	
BSW Parameter		**BSW Type**
FrIfVBTriggeringRef		EcucReferenceDef
BSW Description		
Reference to the assigned Frame triggering.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
FrIf	FrIf/FrIfConfig/FrIfCluster/FrIfController	
BSW Parameter		**BSW Type**
FrIfTransceiver		EcucParamConfContainerDef
BSW Description		
Up to two FlexRay Transceivers may connect a Controller to a Cluster. This container realizes a Controller-Transceiver assignment.		
M2 Template	**M2 Description**	
ECU Resource Template		
M2 Parameter		
Mapping Rule		**Mapping Type**
		unknown

BSW Module	BSW Context	
FrIf	FrIf/FrIfConfig/FrIfCluster/FrIfController/FrIfTransceiver	
BSW Parameter		**BSW Type**
FrIfClusterChannel		EcucEnumerationParamDef
BSW Description		
This parameter identifies to which one of the two Channels (A, B, A and B) of the Cluster the Transceiver is connected. FrIfClusterChannel shall map to Fr_ChannelType: FRIF_CHANNEL_A == FR_CHANNEL_A FRIF_CHANNEL_B == FR_CHANNEL_B FR_CHANNEL_AB shall not be used.		
M2 Template	**M2 Description**	
ECU Resource Template		
M2 Parameter		
Mapping Rule		**Mapping Type**
		unknown

BSW Module	BSW Context	
FrIf	FrIf/FrIfConfig/FrIfCluster/FrIfController/FrIfTransceiver	
BSW Parameter		**BSW Type**
FrIfFrTrcvChannelRef		EcucSymbolicNameReferenceDef
BSW Description		
Reference to a Transceiver Driver Channel. This reference is unique for the ECU.		

M2 Template	M2 Description	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
FrIf	FrIf/FrIfConfig/FrIfCluster	
BSW Parameter		**BSW Type**
FrIfDetectNITError		EcucBooleanParamDef
BSW Description		
Indicates whether NIT error status of each cluster shall be detected or not.		
M2 Template	**M2 Description**	
SystemTemplate	Indicates whether NIT error status of each cluster shall be detected or not.	
M2 Parameter		
Fibex:Fibex4FlexRay::FlexRayCluster.detectNitError		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
FrIf	FrIf/FrIfConfig/FrIfCluster	
BSW Parameter		**BSW Type**
FrIfGChannels		EcucEnumerationParamDef
BSW Description		
The channels that are used by the cluster. Implementation Type: Fr_ChannelType		
M2 Template	**M2 Description**	
System Template	A physical channel is the transmission medium that is used to send and receive information between two communicating ECUs.	
M2 Parameter		
FibexCore::CoreTopology:PhysicalChannel		
Mapping Rule		**Mapping Type**
The channels that are used by the cluster are described in the System Template by the CommunicationCluster-PhysicalChannel relationship.		full

BSW Module	BSW Context	
FrIf	FrIf/FrIfConfig/FrIfCluster	
BSW Parameter		**BSW Type**
FrIfGColdStartAttempts		EcucIntegerParamDef
BSW Description		
Maximum number of times a node in the cluster is permitted to attempt to start the cluster by initiating schedule synchronization		
M2 Template	**M2 Description**	
System Template	The maximum number of times that a node in this cluster is permitted to attempt to start the cluster by initiating schedule synchronization	
M2 Parameter		
Fibex4FlexRay::FlexRayCluster.coldStartAttempts		
Mapping Rule		**Mapping Type**
1:1 mapping		full

— AUTOSAR CONFIDENTIAL —

BSW Module	BSW Context	
FrIf	FrIf/FrIfConfig/FrIfCluster	
BSW Parameter		**BSW Type**
FrIfGCycleCountMax		EcucIntegerParamDef
BSW Description		
Maximum cycle counter value in a given cluster. Remark: Set to 63 for FlexRay Protocol 2.1 Rev. A compliance.		
M2 Template	**M2 Description**	
System Template	Maximum cycle counter value in a given cluster.	
M2 Parameter		
Fibex4FlexRay::FlexRayCluster.cycleCountMax		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
FrIf	FrIf/FrIfConfig/FrIfCluster	
BSW Parameter		**BSW Type**
FrIfGListenNoise		EcucIntegerParamDef
BSW Description		
Upper limit for the start up listen timeout and wake up listen timeout in the presence of noise. It is used as a multiplier of the node parameter pdListenTimeout.		
M2 Template	**M2 Description**	
System Template	Upper limit for the start up and wake up listen timeout in the presence of noise. Expressed as a multiple of the cluster constant pdListenTimeout. Unit microticks	
M2 Parameter		
Fibex4FlexRay::FlexRayCluster.listenNoise		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
FrIf	FrIf/FrIfConfig/FrIfCluster	
BSW Parameter		**BSW Type**
FrIfGMacroPerCycle		EcucIntegerParamDef
BSW Description		
Number of macroticks in a communication cycle. Note: Lower limit 10 for FlexRay Protocol 2.1 Rev. A compliance		
M2 Template	**M2 Description**	
System Template	The number of macroticks in a communication cycle	
M2 Parameter		
Fibex4FlexRay::FlexRayCluster.macroPerCycle		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
FrIf	FrIf/FrIfConfig/FrIfCluster	
BSW Parameter		**BSW Type**
FrIfGMaxWithoutClockCorrectFatal		EcucIntegerParamDef
BSW Description		

Threshold used for testing the vClockCorrectionFailed counter. Defines the number of consecutive even/odd Cycle pairs with missing clock correction terms that will cause the protocol to transition from the POC:normal active or POC:normal passive state into the POC:halt state. [Even/odd cycle pairs].		
M2 Template	**M2 Description**	
System Template	Threshold concerning vClockCorrectionFailedCounter. Defines the number of consecutive even/odd Cycle pairs with missing clock correction terms that will cause the protocol to transition from the POC:normal active or POC:normal passive state into the POC:h	
M2 Parameter		
Fibex4FlexRay::FlexRayCluster.maxWithoutClockCorrectionFatal		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
FrIf	FrIf/FrIfConfig/FrIfCluster	
BSW Parameter		**BSW Type**
FrIfGMaxWithoutClockCorrectPassive		EcucIntegerParamDef
BSW Description		
Threshold used for testing the vClockCorrectionFailed counter. Defines the number of consecutive even/odd Cycle pairs with missing clock correction terms that will cause the protocol to transition from the POC:normal active state to the POC:normal passive state. [Even/Odd cycle pairs]		
M2 Template	**M2 Description**	
System Template	Threshold concerning vClockCorrectionFailedCounter. Defines the number of consecutive even/odd Cycle pairs with missing clock correction terms that will cause the protocol to transition from the POC:normal active state to the POC:normal passive state.	
M2 Parameter		
Fibex4FlexRay::FlexRayCluster.maxWithoutClockCorrectionPassive		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
FrIf	FrIf/FrIfConfig/FrIfCluster	
BSW Parameter		**BSW Type**
FrIfGNetworkManagementVectorLength		EcucIntegerParamDef
BSW Description		
Length of the Network Management vector in a cluster [bytes]		
M2 Template	**M2 Description**	
System Template	Length of the Network Management vector on a cluster. Unit: Bytes	
M2 Parameter		
Fibex4FlexRay::FlexRayCluster.networkManagementVectorLength		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
FrIf	FrIf/FrIfConfig/FrIfCluster	
BSW Parameter		**BSW Type**
FrIfGNumberOfMinislots		EcucIntegerParamDef
BSW Description		

Number of minislots in the dynamic segment Remark: Upper limit 7986 for FlexRay Protocol 2.1 Rev. A compliance	
M2 Template	**M2 Description**
System Template	Number of Minislots in the dynamic segment.
M2 Parameter	
Fibex4FlexRay::FlexRayCluster.numberOfMinislots	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	**BSW Context**
FrIf	FrIf/FrIfConfig/FrIfCluster
BSW Parameter	**BSW Type**
FrIfGNumberOfStaticSlots	EcucIntegerParamDef
BSW Description	
Number of static slots in the static segment	
M2 Template	**M2 Description**
System Template	The number of static slots in the static segment.
M2 Parameter	
Fibex4FlexRay::FlexRayCluster.numberOfStaticSlots	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	**BSW Context**
FrIf	FrIf/FrIfConfig/FrIfCluster
BSW Parameter	**BSW Type**
FrIfGPayloadLengthStatic	EcucIntegerParamDef
BSW Description	
Payload length of a static frame [16 bit words]	
M2 Template	**M2 Description**
System Template	Globally configured payload length of a static frame. Unit: 16-bit WORDS.
M2 Parameter	
Fibex4FlexRay::FlexRayCluster.payloadLengthStatic	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	**BSW Context**
FrIf	FrIf/FrIfConfig/FrIfCluster
BSW Parameter	**BSW Type**
FrIfGSyncFrameIDCountMax	EcucIntegerParamDef
BSW Description	
Maximum number of distinct syncframe identifiers present in a given cluster. This parameter maps to FlexRay Protocol 2.1 Rev. A parameter gSyncNodeMax.	
M2 Template	**M2 Description**
System Template	Maximum number of distinct syncframe identifiers present in a given cluster.
M2 Parameter	
Fibex4FlexRay::FlexRayCluster.syncFrameIdCountMax	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	BSW Context	
FrIf	FrIf/FrIfConfig/FrIfCluster	
BSW Parameter		**BSW Type**
FrIfGdActionPointOffset		EcucIntegerParamDef
BSW Description		
Number of macroticks the action point is offset from the beginning of a static slot.		
M2 Template	**M2 Description**	
System Template	The offset of the action point in networks	
M2 Parameter		
Fibex4FlexRay::FlexRayCluster.actionPointOffset		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
FrIf	FrIf/FrIfConfig/FrIfCluster	
BSW Parameter		**BSW Type**
FrIfGdBit		EcucEnumerationParamDef
BSW Description		
Nominal bit time in seconds		
M2 Template	**M2 Description**	
System Template	Nominal bit time (= 1 / fx:SPEED). gdBit = cSamplesPerBit * gdSampleClock-Period. Unit: seconds (gdBit)	
M2 Parameter		
Fibex4FlexRay::FlexRayCluster.nominalBitTime		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
FrIf	FrIf/FrIfConfig/FrIfCluster	
BSW Parameter		**BSW Type**
FrIfGdCasRxLowMax		EcucIntegerParamDef
BSW Description		
Upper limit of the CAS acceptance windows [gdBit] Remark: Range 67 to 99 for FlexRay Protocol 2.1 Rev. A compliance		
M2 Template	**M2 Description**	
System Template	Upper limit of the Collision Avoidance Symbol (CAS) acceptance window. Unit:bitDuration	
M2 Parameter		
Fibex4FlexRay::FlexRayCluster.casRxLowMax		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
FrIf	FrIf/FrIfConfig/FrIfCluster	
BSW Parameter		**BSW Type**
FrIfGdCycle		EcucFloatParamDef
BSW Description		
Length of the cycle, expressed in [s] Remark: Lower limit 0.000024 for FlexRay Protocol 3.0 compliance.		

M2 Template	M2 Description	
System Template	Length of the cycle. Unit: seconds	
M2 Parameter		
Fibex4FlexRay::FlexRayCluster.cycle		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context		
FrIf	FrIf/FrIfConfig/FrIfCluster		
BSW Parameter		**BSW Type**	
FrIfGdDynamicSlotIdlePhase		EcucIntegerParamDef	
BSW Description			
Duration of the idle phase within a dynamic slot [Minislots].			
M2 Template	**M2 Description**		
System Template	The duration of the dynamic slot idle phase in minislots.		
M2 Parameter			
Fibex4FlexRay::FlexRayCluster.dynamicSlotIdlePhase			
Mapping Rule			**Mapping Type**
1:1 mapping			full

BSW Module	BSW Context		
FrIf	FrIf/FrIfConfig/FrIfCluster		
BSW Parameter		**BSW Type**	
FrIfGdIgnoreAfterTx		EcucIntegerParamDef	
BSW Description			
Duration for which the bitstrobing is paused after transmission [gdBit]. Remark: Set to 0 for FlexRay Protocol 2.1 Rev. A compliance.			
M2 Template	**M2 Description**		
System Template	Duration for which the bitstrobing is paused after transmission [gdBit].		
M2 Parameter			
Fibex4FlexRay::FlexRayCluster.ignoreAfterTx			
Mapping Rule			**Mapping Type**
1:1 mapping			full

BSW Module	BSW Context		
FrIf	FrIf/FrIfConfig/FrIfCluster		
BSW Parameter		**BSW Type**	
FrIfGdMacrotick		EcucFloatParamDef	
BSW Description			
Duration of the cluster wide nominal macrotick, expressed in s			
M2 Template	**M2 Description**		
System Template	Duration of the cluster wide nominal macrotick, expressed in seconds		
M2 Parameter			
Fibex4FlexRay::FlexRayCluster.macrotickDuration			
Mapping Rule			**Mapping Type**
1:1 mapping			full

BSW Module	BSW Context

FrIf	FrIf/FrIfConfig/FrIfCluster	
BSW Parameter		**BSW Type**
FrIfGdMiniSlotActionPointOffset		EcucIntegerParamDef
BSW Description		
Number of Macroticks the Minislot action point is offset from the beginning of a Minislot [Macroticks].		
M2 Template	**M2 Description**	
System Template	The Offset of the action point within a minislot. Unit: macroticks	
M2 Parameter		
Fibex4FlexRay::FlexRayCluster.miniSlotActionPointOffset		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
FrIf	FrIf/FrIfConfig/FrIfCluster	
BSW Parameter		**BSW Type**
FrIfGdMinislot		EcucIntegerParamDef
BSW Description		
Duration of a minislot [Macroticks]		
M2 Template	**M2 Description**	
System Template	The duration of a minislot (dynamic segment). Unit: macroticks.	
M2 Parameter		
Fibex4FlexRay::FlexRayCluster.minislotDuration		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
FrIf	FrIf/FrIfConfig/FrIfCluster	
BSW Parameter		**BSW Type**
FrIfGdNit		EcucIntegerParamDef
BSW Description		
Duration of the Network Idle Time [Macroticks] Remark: Upper limit 805 for FlexRay Protocol 2.1 Rev. A compliance.		
M2 Template	**M2 Description**	
System Template	The duration of the network idle time in macroticks	
M2 Parameter		
Fibex4FlexRay::FlexRayCluster.networkIdleTime		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
FrIf	FrIf/FrIfConfig/FrIfCluster	
BSW Parameter		**BSW Type**
FrIfGdSampleClockPeriod		EcucEnumerationParamDef
BSW Description		
Sample clock period		
M2 Template	**M2 Description**	
System Template	Sample clock period. Unit: seconds	
M2 Parameter		
Fibex4FlexRay::FlexRayCluster.sampleClockPeriod		

— AUTOSAR CONFIDENTIAL —

Mapping Rule	Mapping Type
1:1 mapping	full

BSW Module	BSW Context	
FrIf	FrIf/FrIfConfig/FrIfCluster	
BSW Parameter		**BSW Type**
FrIfGdStaticSlot		EcucIntegerParamDef
BSW Description		
Duration of a static slot [Macroticks]. Remark: Range 4-661 for FlexRay Protocol 2.1 Rev. A compliance.		
M2 Template	**M2 Description**	
System Template	The duration of a slot in the static segment. Unit: macroticks	
M2 Parameter		
Fibex4FlexRay::FlexRayCluster.staticSlotDuration		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
FrIf	FrIf/FrIfConfig/FrIfCluster	
BSW Parameter		**BSW Type**
FrIfGdSymbolWindow		EcucIntegerParamDef
BSW Description		
Duration of the symbol window [Macroticks]. Remark: Range 0-142 for FlexRay Protocol 2.1 Rev. A compliance.		
M2 Template	**M2 Description**	
System Template	The duration of the symbol window. Unit: macroticks	
M2 Parameter		
Fibex4FlexRay::FlexRayCluster.symbolWindow		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
FrIf	FrIf/FrIfConfig/FrIfCluster	
BSW Parameter		**BSW Type**
FrIfGdSymbolWindowActionPointOffset		EcucIntegerParamDef
BSW Description		
Number of macroticks the action point offset is from the beginning of the symbol window [Macroticks]. Remark: Set to GdActionPointOffset for FlexRay Protocol 2.1 Rev. A compliance.		
M2 Template	**M2 Description**	
System Template	Number of macroticks the action point offset is from the beginning of the symbol window [Macroticks].	
M2 Parameter		
Fibex4FlexRay::FlexRayCluster.symbolWindowActionPointOffset		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context

FrIf	FrIf/FrIfConfig/FrIfCluster	
BSW Parameter		**BSW Type**
FrIfGdTSSTransmitter		EcucIntegerParamDef
BSW Description		
Number of bits in the Transmission Start Sequence [gdBits]. Remark: Lower limit 3 for FlexRay Protocol 2.1 Rev. A compliance.		
M2 Template	**M2 Description**	
System Template	Number of bits in the Transmission Start Sequence [gdBits].	
M2 Parameter		
Fibex4FlexRay::FlexRayCluster.transmissionStartSequenceDuration		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
FrIf	FrIf/FrIfConfig/FrIfCluster	
BSW Parameter		**BSW Type**
FrIfGdWakeupRxIdle		EcucIntegerParamDef
BSW Description		
Number of bits used by the node to test the duration of the 'idle' or HIGH phase of a received wakeup [gdBit]. Remarks: This parameter maps to FlexRay Protocol 2.1 Rev. A parameter gdWakeupSymbolRxIdle. Lower limit 14 for FlexRay Protocol 2.1 Rev. A compliance.		
M2 Template	**M2 Description**	
System Template	Number of bits used by the node to test the duration of the 'idle' or HIGH phase of a received wakeup [gdBit].	
M2 Parameter		
Fibex4FlexRay::FlexRayCluster.wakeUpRxIdle		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
FrIf	FrIf/FrIfConfig/FrIfCluster	
BSW Parameter		**BSW Type**
FrIfGdWakeupRxLow		EcucIntegerParamDef
BSW Description		
Number of bits used by the node to test the duration of the LOW phase of a received wakeup [gdBit]. Remarks: This parameter maps to FlexRay Protocol 2.1 Rev. A parameter gdWakeupSymbolRxLow. Lower limit 11 for FlexRay Protocol 2.1 Rev. A compliance.		
M2 Template	**M2 Description**	
System Template	Number of bits used by the node to test the duration of the LOW phase of a received wakeup. Unit:bitDuration	
M2 Parameter		
Fibex4FlexRay::FlexRayCluster.wakeUpRxLow		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
FrIf	FrIf/FrIfConfig/FrIfCluster	
BSW Parameter		**BSW Type**
FrIfGdWakeupRxWindow		EcucIntegerParamDef
BSW Description		

The size of the window used to detect wakeups [gdBit]. Remarks: This parameter maps to FlexRay Protocol 2.1 Rev. A parameter gdWakeupSymbolRxWindow. Upper limit 301 for FlexRay Protocol 2.1 Rev. A compliance.		
M2 Template	**M2 Description**	
System Template	The size of the window used to detect wakeups [gdBit].	
M2 Parameter		
Fibex4FlexRay::FlexRayCluster.wakeUpRxWindow		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
FrIf	FrIf/FrIfConfig/FrIfCluster	
BSW Parameter		**BSW Type**
FrIfGdWakeupTxActive		EcucIntegerParamDef
BSW Description		
Number of bits used by the node to transmit the LOW phase of awakeup symbol and the HIGH and LOW phases of a WUDOP [gdBit]. Remarks: This parameter maps to FlexRay Protocol 2.1 Rev. A parameter gdWakeupSymbolTxLow.		
M2 Template	**M2 Description**	
System Template	Number of bits used by the node to transmit the LOW phase of awakeup symbol and the HIGH and LOW phases of a WUDOP. Unit:bitDuration	
M2 Parameter		
Fibex4FlexRay::FlexRayCluster.wakeUpTxActive		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
FrIf	FrIf/FrIfConfig/FrIfCluster	
BSW Parameter		**BSW Type**
FrIfGdWakeupTxIdle		EcucIntegerParamDef
BSW Description		
Number of bits used by the node to transmit the 'idle' part of a wakeup symbol [gdBit]. Remarks: This parameter maps to FlexRay Protocol 2.1 Rev. A parameter gdWakeupSymbolTxIdle.		
M2 Template	**M2 Description**	
System Template	Number of bits used by the node to transmit the idle part of a wake up symbol. Unit: gDbit	
M2 Parameter		
Fibex4FlexRay::FlexRayCluster.wakeUpTxIdle		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
FrIf	FrIf/FrIfConfig/FrIfCluster	
BSW Parameter		**BSW Type**
FrIfJobList		EcucParamConfContainerDef
BSW Description		

This container specifies a list of all FlexRay Jobs of the Cluster to be performed by FrIf_JobListExec_<ClstIdx>().	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**
FrIf	FrIf/FrIfConfig/FrIfCluster/FrIfJobList
BSW Parameter	**BSW Type**
FrIfAbsTimerRef	EcucSymbolicNameReferenceDef
BSW Description	
Reference to the absolute timer to be used to trigger the interrupt whose ISR contains the FrIf_JobListExec_<ClstIdx>() function.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**
FrIf	FrIf/FrIfConfig/FrIfCluster/FrIfJobList
BSW Parameter	**BSW Type**
FrIfJob	EcucParamConfContainerDef
BSW Description	
A job may contain more than one operation that are executed at a specific point in time.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**
FrIf	FrIf/FrIfConfig/FrIfCluster/FrIfJobList/FrIfJob
BSW Parameter	**BSW Type**
FrIfCommunicationOperation	EcucParamConfContainerDef
BSW Description	
A separate operation which is part of a FlexRay Job and defines what type of action is executed.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

— AUTOSAR CONFIDENTIAL —

BSW Module	BSW Context	
FrIf	FrIf/FrIfConfig/FrIfCluster/FrIfJobList/FrIfJob/FrIfCommunicationOperation	
BSW Parameter		**BSW Type**
FrIfCommunicationAction		EcucEnumerationParamDef
BSW Description		
The action to be performed in the FlexRay Operation		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
FrIf	FrIf/FrIfConfig/FrIfCluster/FrIfJobList/FrIfJob/FrIfCommunicationOperation	
BSW Parameter		**BSW Type**
FrIfCommunicationOperationIdx		EcucIntegerParamDef
BSW Description		
For each FlexRay Communication Job, this index spans a range of zero-based consecutive values and thus defines the order of the FlexRay Communication Operation in the respective FlexRay Communication Job.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
FrIf	FrIf/FrIfConfig/FrIfCluster/FrIfJobList/FrIfJob/FrIfCommunicationOperation	
BSW Parameter		**BSW Type**
FrIfLPduIdxRef		EcucReferenceDef
BSW Description		
Reference to a L-PDu index		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
FrIf	FrIf/FrIfConfig/FrIfCluster/FrIfJobList/FrIfJob/FrIfCommunicationOperation	
BSW Parameter		**BSW Type**
FrIfRxComOpMaxLoop		EcucIntegerParamDef
BSW Description		
Defines the maximum number of loops for the receive RECEIVE_AND_INDICATE (Use case: emptying a FIFO).		
M2 Template	**M2 Description**	

— AUTOSAR CONFIDENTIAL —

M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	BSW Context	
FrIf	FrIf/FrIfConfig/FrIfCluster/FrIfJobList/FrIfJob	
BSW Parameter		**BSW Type**
FrIfCycle		EcucIntegerParamDef
BSW Description		
The FlexRay Cycle in which the communication operation will execute this job		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
FrIf	FrIf/FrIfConfig/FrIfCluster/FrIfJobList/FrIfJob	
BSW Parameter		**BSW Type**
FrIfMacrotick		EcucIntegerParamDef
BSW Description		
Macrotick offset in the Cycle [Macrotick]		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
FrIf	FrIf/FrIfConfig/FrIfCluster	
BSW Parameter		**BSW Type**
FrIfMainFunctionPeriod		EcucFloatParamDef
BSW Description		
The execution cycle of the FrIf_MainFunction_<cluster>() in seconds. The FrIf does not require this information but the BSW scheduler, which invokes the cluster main functions, needs it in order to plan its tasks.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
FrIf	FrIf/FrIfConfig/FrIfCluster	
BSW Parameter		**BSW Type**

FrIfMaxIsrDelay	EcucIntegerParamDef
BSW Description	
The maximum delay in macroticks the FrIf_JoblistExec_<cluster>() function is processed after the absolute timer interrupt was triggered.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**	
FrIf	FrIf/FrIfConfig/FrIfCluster	
BSW Parameter		**BSW Type**
FrIfSafetyMargin		EcucIntegerParamDef
BSW Description		
Additional timespan in macroticks which takes jitter into account to be able to set the JobListPointer to the next possible job which can be executed in case the FlexRay Job List Execution Function has be resynchronized.		
M2 Template	**M2 Description**	
SystemTemplate	Additional timespan in macroticks which takes jitter into account to be able to set the JobListPointer to the next possible job which can be executed in case the FlexRay Job List Execution Function has be resynchronized.	
M2 Parameter		
ibex4FlexRay:FlexRayCluster:safetyMargin		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
FrIf	FrIf/FrIfConfig	
BSW Parameter		**BSW Type**
FrIfFrameStructure		EcucParamConfContainerDef
BSW Description		
The Frame structure specifies a Construction Plan how a Frame is assembled with PDUs and their respective Update-Bits.		
M2 Template	**M2 Description**	
System Template	Data frame which is sent over a communication medium. This element describes the pure Layout of a frame sent on a channel.	
M2 Parameter		
FibexCore::CoreCommunication::Communication::Frame		
Mapping Rule		**Mapping Type**
Create container for each FlexRay Frame that is transmitted or received by the regarded ECU. IPduToFrameMapping element in the System Template contains the construction plan.		full

BSW Module	**BSW Context**	
FrIf	FrIf/FrIfConfig/FrIfFrameStructure	
BSW Parameter		**BSW Type**
FrIfByteOrder		EcucEnumerationParamDef
BSW Description		

This parameter defines the ByteOrder of all Pdus that are mapped into the Frame. The absolute position of a Pdu in the Frame is determined by the definition of the ByteOrder parameter: If BIG_ENDIAN is specified, the FrIfPduOffset indicates the position of the most significant bit in the Frame. If LITTLE_ENDIAN is specified, the FrIfPduOffset indicates the position of the least significant bit in the Frame.	
M2 Template	**M2 Description**
System Template	This attribute defines the order of the bytes of the segment and the packing into the MultiplexedIPdu.
M2 Parameter	
FibexCore::DynamicPart.segmentByteOrder and FibexCore::StaticPart.segmentByteOrder and FibexCore::MultiplexedIPdu.selectorFieldByteOrder	
Mapping Rule	**Mapping Type**
A mix between Little Endian and Big Endian within a MultiplexedIPdu is not allowed.	full

BSW Module	**BSW Context**	
FrIf	FrIf/FrIfConfig/FrIfFrameStructure	
BSW Parameter		**BSW Type**
FrIfPdusInFrame		EcucParamConfContainerDef
BSW Description		
This container holds all the information about a PDU in a FlexRay Frame.		
M2 Template	**M2 Description**	
System Template	A PduToFrameMapping defines the composition of Pdus in each frame.	
M2 Parameter		
FibexCore::CoreCommunication::Communication::Frame::PduToFrameMapping		
Mapping Rule		**Mapping Type**
Container must be created for each IPduToFrameMapping element inside the frame.		full

BSW Module	**BSW Context**	
FrIf	FrIf/FrIfConfig/FrIfFrameStructure/FrIfPdusInFrame	
BSW Parameter		**BSW Type**
FrIfPduOffset		EcucIntegerParamDef
BSW Description		
The value specifies the offset of the PDU within the Frame [bytes].		
M2 Template	**M2 Description**	
System Template	This parameter is necessary to describe the byteposition of a Pdu within a Frame.	
M2 Parameter		
CoreCommunication:Communication:Frame:PduToFrameMapping.startPosition		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
FrIf	FrIf/FrIfConfig/FrIfFrameStructure/FrIfPdusInFrame	
BSW Parameter		**BSW Type**
FrIfPduRef		EcucReferenceDef
BSW Description		

This is the reference to the local definition of a PDU.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**
FrIf	FrIf/FrIfConfig/FrIfFrameStructure/FrIfPdusInFrame
BSW Parameter	**BSW Type**
FrIfPduUpdateBitOffset	EcucIntegerParamDef
BSW Description	
This value specifies where the PDU's Update-Bit is stored in the Frame (bit location of PDU's Update-Bit in the FlexRay Frame).	
M2 Template	**M2 Description**
System Template	Indication to the receivers that the corresponding I-Pdu was updated by the sender. This attribute describes the position of the update bit in the frame that aggregates this PDUToFrameMapping. Length is always one bit.
M2 Parameter	
CoreCommunication:Communication:Frame:PduToFrameMapping.updateIndicationBitPosition	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	**BSW Context**
FrIf	FrIf/FrIfConfig
BSW Parameter	**BSW Type**
FrIfPdu	EcucParamConfContainerDef
BSW Description	
Contains PDU information. A PDU may be either a transmission PDU or a reception PDU.	
M2 Template	**M2 Description**
System Template	An IPdu, NmPdu or NPdu (XOR).
M2 Parameter	
CoreCommunication::Pdu	
Mapping Rule	**Mapping Type**
The container must be created for each Pdu that is contained in a FlexRay Frame.	full

BSW Module	**BSW Context**
FrIf	FrIf/FrIfConfig/FrIfPdu
BSW Parameter	**BSW Type**
FrIfPduDirection	EcucChoiceContainerDef
BSW Description	
A PDU is either transmit or receive	
M2 Template	**M2 Description**
System Template	Communication Direction of the Connector Port (input or output Port).
M2 Parameter	
CoreCommunication::FrameTriggering::FramePort.communicationDirection	
Mapping Rule	**Mapping Type**

The container must be created for each Pdu that is transmitted or received in a FlexRay Frame.	full

BSW Module	BSW Context		
FrIf	FrIf		
BSW Parameter		**BSW Type**	
FrIfGeneral		EcucParamConfContainerDef	
BSW Description			
This container contains the general configuration parameters of the FlexRay Interface.			
M2 Template	**M2 Description**		
System Template	The CommunicationCluster is the main element to describe the topological connection of communicating ECUs.		
M2 Parameter			
Fibex4FlexRay::FlexRayCluster			
Mapping Rule			**Mapping Type**
Container must be created if the ECU is connected to a FlexRay Cluster			full

BSW Module	BSW Context		
FrIf	FrIf/FrIfGeneral		
BSW Parameter		**BSW Type**	
FrIfAbsTimerIdx		EcucIntegerParamDef	
BSW Description			
Maximum number of supported absolut timers.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	BSW Context		
FrIf	FrIf/FrIfGeneral		
BSW Parameter		**BSW Type**	
FrIfAllSlotsSupport		EcucBooleanParamDef	
BSW Description			
Configuration parameter to enable/disable FrIf support to enable/disable of switching from key-slot / single-slot mode to all slot mode.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	BSW Context	
FrIf	FrIf/FrIfGeneral	
BSW Parameter		**BSW Type**
FrIfCancelTransmitSupport		EcucBooleanParamDef
BSW Description		

Configuration parameter to enable/disable FrIf support to request the cancellation of the I-PDU transmission to FrDrv.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**		
FrIf	FrIf/FrIfGeneral		
BSW Parameter		**BSW Type**	
FrIfDevErrorDetect		EcucBooleanParamDef	
BSW Description			
Switches the Development Error Detection and Notification on or off true: Development Error Detection and Notification on false: Development Error Detection and Notification off			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	**BSW Context**		
FrIf	FrIf/FrIfGeneral		
BSW Parameter		**BSW Type**	
FrIfDisableLPduSupport		EcucBooleanParamDef	
BSW Description			
Configuration parameter to enable/disable FrIf support to disables the hardware resource of a LPdu for transmission/reception.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	**BSW Context**		
FrIf	FrIf/FrIfGeneral		
BSW Parameter		**BSW Type**	
FrIfDisableTransceiverBranchSupport		EcucBooleanParamDef	
BSW Description			
Configuration parameter to enable/disable FrIf support to disable branches of an active star.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**

	local

BSW Module	BSW Context	
FrIf	FrIf/FrIfGeneral	
BSW Parameter		**BSW Type**
FrIfEnableTransceiverBranchSupport		EcucBooleanParamDef
BSW Description		
Configuration parameter to enable/disable FrIf support to enable branches of an active star.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
FrIf	FrIf/FrIfGeneral	
BSW Parameter		**BSW Type**
FrIfGetClockCorrectionSupport		EcucBooleanParamDef
BSW Description		
Configuration parameter to enable/disable FrIf support to enable/disable of polling the FlexRay Driver to getting CC clock correction values.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
FrIf	FrIf/FrIfGeneral	
BSW Parameter		**BSW Type**
FrIfGetGetChannelStatusSupport		EcucBooleanParamDef
BSW Description		
Configuration parameter to enable/disable FrIf support to enable/disable of polling the FlexRay Driver to getting error information about the FlexRay communications bus.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
FrIf	FrIf/FrIfGeneral	
BSW Parameter		**BSW Type**
FrIfGetNmVectorSupport		EcucBooleanParamDef
BSW Description		
Configuration parameter to enable/disable FrIf support to request the FlexRay hardware NMVector.		

M2 Template	M2 Description	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
FrIf	FrIf/FrIfGeneral	
BSW Parameter		**BSW Type**
FrIfGetNumOfStartupFramesSupport		EcucBooleanParamDef
BSW Description		
Configuration parameter to enable/disable FrIf support to enable/disable of polling the FlexRay Driver for the actual number of received startup frames on the bus.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
FrIf	FrIf/FrIfGeneral	
BSW Parameter		**BSW Type**
FrIfGetSyncFrameListSupport		EcucBooleanParamDef
BSW Description		
Configuration parameter to enable/disable FrIf support to enable/disable of polling the FlexRay Driver to getting a list of actual received sync frames.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
FrIf	FrIf/FrIfGeneral	
BSW Parameter		**BSW Type**
FrIfGetTransceiverErrorSupport		EcucBooleanParamDef
BSW Description		
Configuration parameter to enable/disable FrIf support to get the FlexRay Transceiver errors by calling the FlexRay Transceiver module.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

— AUTOSAR CONFIDENTIAL —

BSW Module	BSW Context	
FrIf	FrIf/FrIfGeneral	
BSW Parameter		**BSW Type**
FrIfGetWakeupRxStatusSupport		EcucBooleanParamDef
BSW Description		
Configuration parameter to enable/disable FrIf support to get the wakeup received information from the FlexRay controller.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
FrIf	FrIf/FrIfGeneral	
BSW Parameter		**BSW Type**
FrIfNumClstSupported		EcucIntegerParamDef
BSW Description		
Maximum number of FlexRay Clusters that the FlexRay Interface supports.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
FrIf	FrIf/FrIfGeneral	
BSW Parameter		**BSW Type**
FrIfNumCtrlSupported		EcucIntegerParamDef
BSW Description		
Maximum number of FlexRay CCs that the FlexRay Interface supports		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
FrIf	FrIf/FrIfGeneral	
BSW Parameter		**BSW Type**
FrIfPublicCddHeaderFile		EcucStringParamDef
BSW Description		
Defines header files for callback functions which shall be included in case of CDDs. Range of characters is 1.. 32.		
M2 Template	**M2 Description**	
M2 Parameter		

Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**	
FrIf	FrIf/FrIfGeneral	
BSW Parameter		**BSW Type**
FrIfReadCCConfigApi		EcucBooleanParamDef
BSW Description		
Configuration parameter to enable/disable the optional FrIf_ReadCCConfig API.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
FrIf	FrIf/FrIfGeneral	
BSW Parameter		**BSW Type**
FrIfReconfigLPduSupport		EcucBooleanParamDef
BSW Description		
Configuration parameter to enable/disable FrIf support to enable/disable the reconfiguration of a given LPdu according to the parameters (FrameId, Channel, CycleRepetition, CycleOffset, PayloadLength, HeaderCRC) at runtime.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
FrIf	FrIf/FrIfGeneral	
BSW Parameter		**BSW Type**
FrIfUnusedBitValue		EcucIntegerParamDef
BSW Description		
Set unused bits to a defined value.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
FrIf	FrIf/FrIfGeneral	
BSW Parameter		**BSW Type**
FrIfVersionInfoApi		EcucBooleanParamDef

BSW Description		
Enables/disables the existence of the FrIf_GetVersionInfo() API service true: FrIf_GetVersionInfo() API service exists false: FrIf_GetVersionInfo() API service does not exist		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

D.8 FlexRayNm Mapping

BSW Module	BSW Context		
FrNm	FrNm		
BSW Parameter		**BSW Type**	
FrNmChannelConfig		EcucParamConfContainerDef	
BSW Description			
This container contains the configuration parameters for all FlexRay NM channels.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	BSW Context		
FrNm	FrNm/FrNmChannelConfig		
BSW Parameter		**BSW Type**	
FrNmChannel		EcucParamConfContainerDef	
BSW Description			
This container contains the configuration parameters for a FlexRay NM Channel.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	BSW Context	
FrNm	FrNm/FrNmChannelConfig/FrNmChannel	
BSW Parameter		**BSW Type**
FrNmChannelIdentifiers		EcucParamConfContainerDef
BSW Description		
This container contains instance specific identifiers related to the respective FlexRay Channel.		
M2 Template	**M2 Description**	
M2 Parameter		

— AUTOSAR CONFIDENTIAL —

Mapping Rule	Mapping Type
	local

BSW Module	BSW Context	
FrNm	FrNm/FrNmChannelConfig/FrNmChannel/FrNmChannelIdentifiers	
BSW Parameter		**BSW Type**
FrNmCarWakeUpBitPosition		EcucIntegerParamDef
BSW Description		
Specifies the Bit position of the CWU within the NM-Message.		
M2 Template	**M2 Description**	
SystemTemplate	Specifies the bit position of the CarWakeUp within the NM-Message.	
M2 Parameter		
NetworkManagement::FlexrayNmCluster.nmCarWakeUpBitPosition		
Mapping Rule		**Mapping Type**
The position of the Car Wakeup bit in the Ecuc is defined by the configuration parameters FrNmCarWakeUpBytePosition and FrNmCarWakeUpBitPosition (position in wakeUpByte). In the SysT the position is described only by the bit position in the NmMessage.		full

BSW Module	BSW Context	
FrNm	FrNm/FrNmChannelConfig/FrNmChannel/FrNmChannelIdentifiers	
BSW Parameter		**BSW Type**
FrNmCarWakeUpBytePosition		EcucIntegerParamDef
BSW Description		
Specifies the Byte position of the CWU within the NM-Message.		
M2 Template	**M2 Description**	
SystemTemplate	Specifies the bit position of the CarWakeUp within the NM-Message.	
M2 Parameter		
NetworkManagement::FlexrayNmCluster.nmCarWakeUpBitPosition		
Mapping Rule		**Mapping Type**
The position of the Car Wakeup bit in the Ecuc is defined by the configuration parameters FrNmCarWakeUpBytePosition and FrNmCarWakeUpBitPosition (position in wakeUpByte). In the SysT the position is described only by the bit position in the NmMessage.		full

BSW Module	BSW Context	
FrNm	FrNm/FrNmChannelConfig/FrNmChannel/FrNmChannelIdentifiers	
BSW Parameter		**BSW Type**
FrNmCarWakeUpFilterEnabled		EcucBooleanParamDef
BSW Description		
If CWU filtering is supported, only the CWU bit within the NM message with source node identifier FrNmCarWakeUpFilterNodeId is considered as CWU request. FALSE - CWU Filtering is not supported TRUE - CWU Filtering is supported		
M2 Template	**M2 Description**	
SystemTemplate	If this attribute is set to true the CareWakeUp filtering is supported. In this case only the CarWakeUp bit within the NM message with source node identifier nmCarWakeUpFilterNodeId is considered as CarWakeUp request.	
M2 Parameter		
NetworkManagement::FlexrayNmCluster.nmCarWakeUpFilterEnabled		
Mapping Rule		**Mapping Type**

<table>
<tr><td colspan="3">1:1 mapping</td><td>full</td></tr>
</table>

<table>
<tr><th>BSW Module</th><th colspan="3">BSW Context</th></tr>
<tr><td>FrNm</td><td colspan="3">FrNm/FrNmChannelConfig/FrNmChannel/FrNmChannelIdentifiers</td></tr>
<tr><th colspan="2">BSW Parameter</th><th colspan="2">BSW Type</th></tr>
<tr><td colspan="2">FrNmCarWakeUpFilterNodeId</td><td colspan="2">EcucIntegerParamDef</td></tr>
<tr><th colspan="4">BSW Description</th></tr>
<tr><td colspan="4">Source node identifier for CWU filtering. If CWU filtering is supported, only the CWU bit within the NM
message with source node identifier FrNmCarWakeUpFilterNodeId is considered as CWU
request.</td></tr>
<tr><th>M2 Template</th><th colspan="3">M2 Description</th></tr>
<tr><td>SystemTemplate</td><td colspan="3">Source node identifier for CarWakeUp filtering. If CarWakeUp filtering is supported (nmCarWakeUpFilterEnabled), only the CarWakeUp bit within the NM message with source node identifier nmCarWakeUpFilterNodeId is considered as CarWakeUp request.</td></tr>
<tr><th colspan="4">M2 Parameter</th></tr>
<tr><td colspan="4">NetworkManagement::FlexrayNmCluster.nmCarWakeUpFilterNodeId</td></tr>
<tr><th colspan="3">Mapping Rule</th><th>Mapping Type</th></tr>
<tr><td colspan="3">1:1 mapping</td><td>full</td></tr>
</table>

<table>
<tr><th>BSW Module</th><th colspan="3">BSW Context</th></tr>
<tr><td>FrNm</td><td colspan="3">FrNm/FrNmChannelConfig/FrNmChannel/FrNmChannelIdentifiers</td></tr>
<tr><th colspan="2">BSW Parameter</th><th colspan="2">BSW Type</th></tr>
<tr><td colspan="2">FrNmCarWakeUpRxEnabled</td><td colspan="2">EcucBooleanParamDef</td></tr>
<tr><th colspan="4">BSW Description</th></tr>
<tr><td colspan="4">Enables or disables support of CarWakeUp bit evaluation in received NM messages.
FALSE - CarWakeUp not supported
TRUE - CarWakeUp supported</td></tr>
<tr><th>M2 Template</th><th colspan="3">M2 Description</th></tr>
<tr><td>SystemTemplate</td><td colspan="3">If set to true this attribute enables the support of CarWakeUp bit evaluation in received NM messages.</td></tr>
<tr><th colspan="4">M2 Parameter</th></tr>
<tr><td colspan="4">NetworkManagement::FlexrayNmCluster.nmCarWakeUpRxEnabled</td></tr>
<tr><th colspan="3">Mapping Rule</th><th>Mapping Type</th></tr>
<tr><td colspan="3">1:1 mapping</td><td>full</td></tr>
</table>

<table>
<tr><th>BSW Module</th><th colspan="3">BSW Context</th></tr>
<tr><td>FrNm</td><td colspan="3">FrNm/FrNmChannelConfig/FrNmChannel/FrNmChannelIdentifiers</td></tr>
<tr><th colspan="2">BSW Parameter</th><th colspan="2">BSW Type</th></tr>
<tr><td colspan="2">FrNmChannelHandle</td><td colspan="2">EcucSymbolicNameReferenceDef</td></tr>
<tr><th colspan="4">BSW Description</th></tr>
<tr><td colspan="4">Channel identifier configured for the respective instance of the NM.

The FrNmChannelHandle shall be encoded in the FrNmRxPduId parameter which is passed to FrNm_RxIndication() function called by the FrIf.</td></tr>
<tr><th>M2 Template</th><th colspan="3">M2 Description</th></tr>
<tr><td></td><td colspan="3"></td></tr>
<tr><th colspan="4">M2 Parameter</th></tr>
<tr><td colspan="4"></td></tr>
<tr><th colspan="3">Mapping Rule</th><th>Mapping Type</th></tr>
</table>

	local

BSW Module	BSW Context	
FrNm	FrNm/FrNmChannelConfig/FrNmChannel/FrNmChannelIdentifiers	
BSW Parameter		**BSW Type**
FrNmComMNetworkHandleRef		EcucSymbolicNameReferenceDef
BSW Description		
This reference points to the unique channel defined by the ComMChannel and provides access to the unique channel index value in ComMChannelId.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
FrNm	FrNm/FrNmChannelConfig/FrNmChannel/FrNmChannelIdentifiers	
BSW Parameter		**BSW Type**
FrNmControlBitVectorActive		EcucBooleanParamDef
BSW Description		
This parameter is used to activate or deactivate the control bit vector support for a Fr Nm Channel.		
M2 Template	**M2 Description**	
SystemTemplate	Used to activate or deactivate the control bit vector support for a Fr Nm Channel.	
M2 Parameter		
NetworkManagement::FlexRayNmCluster.nmControlBitVectorActive		
Mapping Rule		**Mapping Type**
full		full

BSW Module	BSW Context	
FrNm	FrNm/FrNmChannelConfig/FrNmChannel/FrNmChannelIdentifiers	
BSW Parameter		**BSW Type**
FrNmNodeId		EcucIntegerParamDef
BSW Description		
NM node identifier configured for the respective FlexRay Channel. It is used for identifying the respective NM node in the NM-cluster. It must be unique for each NM node within one NM cluster.		
M2 Template	**M2 Description**	
SystemTemplate	Node identifier of local NmNode. Must be unique in the NmCluster.	
M2 Parameter		
NetworkManagement::FlexrayNmNode.nmNodeId		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
FrNm	FrNm/FrNmChannelConfig/FrNmChannel/FrNmChannelIdentifiers	
BSW Parameter		**BSW Type**
FrNmPduScheduleVariant		EcucEnumerationParamDef

BSW Description		
This parameter defines the PDU scheduling variant that should be used for this channel. Option 1 NM-Vote and NM-Data in static segment (one PDU) Option 2 NM-Vote and NM-Data in dynamic segment (one PDU) Option 3 NM-Vote and NM-Data in static segment (separate PDU) Option 4 NM-Vote in static segment and NM-Data in dynamic segment Option 5 NM-Vote in dynamic segment and NM-Data in static segment Option 6 NM-Vote and NM-Data in dynamic segment (separate PDU) Option 7 Combined NM-Vote and CBV in static segment and NM-Data in dynamic segment		
M2 Template	**M2 Description**	
SystemTemplate	FrNm schedule variant according to FrNm SWS.	
M2 Parameter		
NetworkManagement::FlexrayNmClusterCoupling.nmScheduleVariant		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
FrNm	FrNm/FrNmChannelConfig/FrNmChannel/FrNmChannelIdentifiers	
BSW Parameter		**BSW Type**
FrNmPnEnabled		EcucBooleanParamDef
BSW Description		
Enables or disables support of partial networking. false: Partial networking Range not supported true: Partial networking supported		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**

BSW Module	BSW Context	
FrNm	FrNm/FrNmChannelConfig/FrNmChannel/FrNmChannelIdentifiers	
BSW Parameter		**BSW Type**
FrNmPnEraCalcEnabled		EcucBooleanParamDef
BSW Description		
Specifies if FrNm calculates the PN request information for external requests. (ERA) false: PN request are not calculated true: PN request are calculated		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**

BSW Module	BSW Context
FrNm	FrNm/FrNmChannelConfig/FrNmChannel/FrNmChannelIdentifiers

BSW Parameter		BSW Type
FrNmPnEraRxNSduRef		EcucReferenceDef
BSW Description		
Reference to a Pdu in the COM-Stack. Only one SduRef is required for FrNm because the EIRA is the aggregation over all FlexRay Channels.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
FrNm	FrNm/FrNmChannelConfig/FrNmChannel/FrNmChannelIdentifiers	
BSW Parameter		**BSW Type**
FrNmRepeatMessageBitActive		EcucBooleanParamDef
BSW Description		
This parameter is used to activate or deactivate the repeat message bit support for a Fr Nm Channel.		
M2 Template	**M2 Description**	
SystemTemplate	Used to activate or deactivate the repeat message bit support for a Fr Nm Channel.	
M2 Parameter		
NetworkManagement::FlexRayNmCluster.nmRepeatMessageBitActive		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
FrNm	FrNm/FrNmChannelConfig/FrNmChannel/FrNmChannelIdentifiers	
BSW Parameter		**BSW Type**
FrNmRxPdu		EcucParamConfContainerDef
BSW Description		
This container describes the FlexRay NM RX PDU:s.		
M2 Template	**M2 Description**	
SystemTemplate	receive NM Pdu	
M2 Parameter		
NetworkManagement::NmNode.rxNmPdu		
Mapping Rule		**Mapping Type**
Create Container if the regarded NmNode recieves a Pdu		full

BSW Module	BSW Context	
FrNm	FrNm/FrNmChannelConfig/FrNmChannel/FrNmChannelIdentifiers/FrNmRxPdu	
BSW Parameter		**BSW Type**
FrNmRxPduContainsData		EcucBooleanParamDef
BSW Description		
This parameted defines if the PDU contains NM Data.		
M2 Template	**M2 Description**	
SystemTemplate	Defines if the PDU contains NM Data.	
M2 Parameter		
NetworkManagement::NmPdu.nmDataInformation		

Mapping Rule	Mapping Type
1:1 mapping	full

BSW Module	BSW Context	
FrNm	FrNm/FrNmChannelConfig/FrNmChannel/FrNmChannelIdentifiers/FrNmRxPdu	
BSW Parameter		**BSW Type**
FrNmRxPduContainsVote		EcucBooleanParamDef
BSW Description		
This parameted defines if the PDU contains NM Vote information.		
M2 Template	**M2 Description**	
SystemTemplate	efines if the PDU contains NM Vote information.	
M2 Parameter		
NetworkManagement::NmPdu.nmVoteInformation		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
FrNm	FrNm/FrNmChannelConfig/FrNmChannel/FrNmChannelIdentifiers/FrNmRxPdu	
BSW Parameter		**BSW Type**
FrNmRxPduId		EcucIntegerParamDef
BSW Description		
PDU identifier configured for the respective FlexRay Channel. It is used for referring to the FlexRay Interface receive function. It must be consistent with the value configured in the FlexRay Interface. This ID is used for the combined reception of NM Vote and NM Data or for the reception of the NM Vote if NM Data is received in a separate PDU. ImplementationType: PduIdType		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
FrNm	FrNm/FrNmChannelConfig/FrNmChannel/FrNmChannelIdentifiers/FrNmRxPdu	
BSW Parameter		**BSW Type**
FrNmRxPduRef		EcucReferenceDef
BSW Description		
The reference to a PDU in the global PDU structure described in the AUTOSAR ECU Configuration Specification. This reference will be used by the FrIf module to derive the PDU Id.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**
FrNm	FrNm/FrNmChannelConfig/FrNmChannel/FrNmChannelIdentifiers
BSW Parameter	**BSW Type**
FrNmSynchronizationPointEnabled	EcucBooleanParamDef
BSW Description	
This parameter defines if this channel shall provide the synchronization point indication to the NM Interface.	
M2 Template	**M2 Description**
SystemTemplate	If this parameter is true, then this network is a synchronizing network for the NM coordination cluster which it belongs to. The network is expected to call Nm_SynchronizationPoint() at regular intervals.
M2 Parameter	
NetworkManagement::NmCluster.nmSynchronizingNetwork	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	**BSW Context**
FrNm	FrNm/FrNmChannelConfig/FrNmChannel/FrNmChannelIdentifiers
BSW Parameter	**BSW Type**
FrNmTxPdu	EcucParamConfContainerDef
BSW Description	
This container describes the FlexRay NM TX PDU:s.	
M2 Template	**M2 Description**
SystemTemplate	transmit NM Pdu
M2 Parameter	
NetworkManagement::NmNode.txNmPdu	
Mapping Rule	**Mapping Type**
Create Container if the regarded NmNode transmits a Pdu	full

BSW Module	**BSW Context**
FrNm	FrNm/FrNmChannelConfig/FrNmChannel/FrNmChannelIdentifiers/FrNmTxPdu
BSW Parameter	**BSW Type**
FrNmTxConfirmationPduId	EcucIntegerParamDef
BSW Description	
Handle Id to be used by the Lower Layer to confirm the transmission of the FrNmTxPdu to the LowerLayer.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**
FrNm	FrNm/FrNmChannelConfig/FrNmChannel/FrNmChannelIdentifiers/FrNmTxPdu
BSW Parameter	**BSW Type**
FrNmTxPduContainsData	EcucBooleanParamDef
BSW Description	
This parameted defines if the PDU contains NM Data.	
M2 Template	**M2 Description**

SystemTemplate	Defines if the PDU contains NM Data.
M2 Parameter	
NetworkManagement::NmPdu.nmDataInformation	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	**BSW Context**
FrNm	FrNm/FrNmChannelConfig/FrNmChannel/FrNmChannelIdentifiers/FrNmTxPdu
BSW Parameter	**BSW Type**
FrNmTxPduContainsVote	EcucBooleanParamDef
BSW Description	
This parameted defines if the PDU contains NM Vote information.	
M2 Template	**M2 Description**
SystemTemplate	Defines if the PDU contains NM Vote information.
M2 Parameter	
NetworkManagement::NmPdu.nmVoteInformation	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	**BSW Context**
FrNm	FrNm/FrNmChannelConfig/FrNmChannel/FrNmChannelIdentifiers/FrNmTxPdu
BSW Parameter	**BSW Type**
FrNmTxPduRef	EcucReferenceDef
BSW Description	
The reference to a PDU in the global PDU structure described in the AUTOSAR ECU Configuration Specification. This reference is used to derive the PDU Id that is defined by the FrIf module.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**
FrNm	FrNm/FrNmChannelConfig/FrNmChannel/FrNmChannelIdentifiers
BSW Parameter	**BSW Type**
FrNmUserDataTxPdu	EcucParamConfContainerDef
BSW Description	
This optional container is used to configure the UserNm PDU. This container is only available if FrNmComUserDataSupport is enabled.	
M2 Template	**M2 Description**
SystemTemplate	This optional aggregation is used to describe NmUserData that is transmitted in the NmPdu.
M2 Parameter	
NetworkManagement::CanNmPdu.iSignalToIPduMapping	
Mapping Rule	**Mapping Type**
Create container for each NmPdu that aggregates the ISignalToIPduMapping element. The configuration for these Pdus (e.g. Transfer Properties) shall be derived from this information.	full

BSW Module	BSW Context	
FrNm	FrNm/FrNmChannelConfig/FrNmChannel/FrNmChannelIdentifiers/FrNmUserDataTxPdu	
BSW Parameter		**BSW Type**
FrNmTxUserDataPduId		EcucIntegerParamDef
BSW Description		
This parameter defines the Handle ID of the NM User Data I-PDU.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
FrNm	FrNm/FrNmChannelConfig/FrNmChannel/FrNmChannelIdentifiers/FrNmUserDataTxPdu	
BSW Parameter		**BSW Type**
FrNmTxUserDataPduRef		EcucReferenceDef
BSW Description		
Reference to the NM User Data I-PDU in the global PDU collection.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
FrNm	FrNm/FrNmChannelConfig/FrNmChannel	
BSW Parameter		**BSW Type**
FrNmChannelTiming		EcucParamConfContainerDef
BSW Description		
This container contains instance-specific timing related to the respective FlexRay Channel.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
FrNm	FrNm/FrNmChannelConfig/FrNmChannel/FrNmChannelTiming	
BSW Parameter		**BSW Type**
FrNmDataCycle		EcucEnumerationParamDef
BSW Description		
Number of FlexRay Schedule Cycles needed to transmit the NM Data of all ECUs on the FlexRay bus		
M2 Template	**M2 Description**	

SystemTemplate	Number of FlexRay Communication Cycles needed to transmit the Nm Data PDUs of all FlexRay Nm Ecus of this FlexRayNmCluster.
M2 Parameter	
NetworkManagementFlexRayNmCluster.nmDataCycle	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	**BSW Context**	
FrNm	FrNm/FrNmChannelConfig/FrNmChannel/FrNmChannelTiming	
BSW Parameter		**BSW Type**
FrNmMainFunctionPeriod		EcucFloatParamDef
BSW Description		
This parameter defines the processing cycle of the main function of FrNm module in seconds.		
M2 Template	**M2 Description**	
SystemTemplate	Defines the processing cycle of the main function of FrNm module.	
M2 Parameter		
NetworkManagement::FlexrayNmCluster.nmMainFunctionPeriod		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
FrNm	FrNm/FrNmChannelConfig/FrNmChannel/FrNmChannelTiming	
BSW Parameter		**BSW Type**
FrNmMsgTimeoutTime		EcucFloatParamDef
BSW Description		
Timeout of a NM-message. It determines in seconds how long the NM shall wait with notification of transmission failure while communication errors occur on the bus.		
M2 Template	**M2 Description**	
SystemTemplate	Timeout of a NM message in seconds. It determines how long the NM shall wait with notification of transmission failure while communication errors occur on the bus.	
M2 Parameter		
NetworkManagement::FlexrayNmCluster.nmMessageTimeoutTime		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
FrNm	FrNm/FrNmChannelConfig/FrNmChannel/FrNmChannelTiming	
BSW Parameter		**BSW Type**
FrNmReadySleepCnt		EcucIntegerParamDef
BSW Description		
Numbers of repetitions in the ready sleep state before NM switches to bus sleep mode. On a value of "1", the NM-State Machine will leave the Ready Sleep State after one NM Repetition Cycle with no "keep awake" votes.		
M2 Template	**M2 Description**	
SystemTemplate	Numbers of repetitions in the ready sleep state before NM switches to bus sleep mode. On a value of "1", the NM-State Machine will leave the Ready Sleep State after one NM Repetition Cycle with no "keep awake" votes.	
M2 Parameter		
NetworkManagement::FlexRayNmCluster.nmReadySleepCount		

Mapping Rule	Mapping Type
1:1 mapping	full

BSW Module	BSW Context	
FrNm	FrNm/FrNmChannelConfig/FrNmChannel/FrNmChannelTiming	
BSW Parameter		**BSW Type**
FrNmRemoteSleepIndTime		EcucFloatParamDef
BSW Description		
Timeout for Remote Sleep Indication. It defines the time in seconds how long it shall take to recognize that all other nodes are ready to sleep. The value "0" denotes that no Remote Sleep Indication functionality is configured.		
M2 Template	**M2 Description**	
SystemTemplate	Timeout for Remote Sleep Indication in seconds. It defines the time how long it shall take to recognize that all other nodes are ready to sleep.	
M2 Parameter		
NetworkManagement::FlexrayNmCluster.nmRemoteSleepIndicationTime		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
FrNm	FrNm/FrNmChannelConfig/FrNmChannel/FrNmChannelTiming	
BSW Parameter		**BSW Type**
FrNmRepeatMessageTime		EcucFloatParamDef
BSW Description		
Timeout for Repeat Message State. Defines the time in seconds how long the NM shall stay in the Repeat Message State. The value "0" denotes that no Repeat Message State is configured, which means that Repeat Message State is transient and implies that it is left immediately after entry and consequently no startup stability is guaranteed and no node detection procedure is possible.		
M2 Template	**M2 Description**	
SystemTemplate	Timeout for Repeat Message State. Defines the time in seconds how long the NM shall stay in the Repeat Message State.	
M2 Parameter		
NetworkManagement::FlexrayNmCluster.nmRepeatMessageTime		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
FrNm	FrNm/FrNmChannelConfig/FrNmChannel/FrNmChannelTiming	
BSW Parameter		**BSW Type**
FrNmRepetitionCycle		EcucEnumerationParamDef
BSW Description		
Number of Flexray Schedule Cycles used to repeat the transmission of the Nm vote of all ECUs on the Flexray Bus.		
M2 Template	**M2 Description**	
SystemTemplate	Number of FlexRay Communication Cycles used to repeat the transmission of the Nm vote PDUs of all FlexRay NmEcus of this FlexRayNmCluster. This value must be an integral multiple of nmVotingCycle.	
M2 Parameter		

— AUTOSAR CONFIDENTIAL —

NetworkManagementFlexRayNmCluster.nmRepetitionCycle	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	**BSW Context**	
FrNm	FrNm/FrNmChannelConfig/FrNmChannel/FrNmChannelTiming	
BSW Parameter		**BSW Type**
FrNmSyncLossTimer		EcucFloatParamDef
BSW Description		
Initial value for the SyncLossTimer in seconds.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**

BSW Module	**BSW Context**	
FrNm	FrNm/FrNmChannelConfig/FrNmChannel/FrNmChannelTiming	
BSW Parameter		**BSW Type**
FrNmVoteInhibitionEnabled		EcucBooleanParamDef
BSW Description		
Pre-processor switch for enabling the inhibition of vote changes from the next-to-last repetition cycle to the last repetition cycle before the Ready Sleep Counter expires.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**

BSW Module	**BSW Context**	
FrNm	FrNm/FrNmChannelConfig/FrNmChannel/FrNmChannelTiming	
BSW Parameter		**BSW Type**
FrNmVotingCycle		EcucEnumerationParamDef
BSW Description		
Number of FlexRay Schedule Cycles needed to transmit the Nm vote of all ECUs on the FlexRay Bus.		
M2 Template	**M2 Description**	
SystemTemplate	Number of FlexRay CommunicationCycles needed to transmit the Nm vote of Pdus of all FlexRay NmEcus of this FlexRayNmCluster.	
M2 Parameter		
NetworkManagementFlexRayNmCluster.nmVotingCycle		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
FrNm	FrNm	
BSW Parameter		**BSW Type**

— AUTOSAR CONFIDENTIAL —

FrNmGlobalConfig		EcucParamConfContainerDef
BSW Description		
This container contains all global configuration parameters for the FrNm module.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
FrNm	FrNm/FrNmGlobalConfig	
BSW Parameter		**BSW Type**
FrNmGlobalConstants		EcucParamConfContainerDef
BSW Description		
This container contains module constants related to the FlexRay NM functionality.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
FrNm	FrNm/FrNmGlobalConfig/FrNmGlobalConstants	
BSW Parameter		**BSW Type**
FrNmNumberOfClusters		EcucIntegerParamDef
BSW Description		
Number of AUTOSAR FR NM clusters allowed within one ECU.		
M2 Template	**M2 Description**	
SystemTemplate	Collection of NM Clusters	
M2 Parameter		
NetworkManagement::NmCluster		
Mapping Rule		**Mapping Type**
Count aggregated NMClusters		full

BSW Module	**BSW Context**	
FrNm	FrNm/FrNmGlobalConfig	
BSW Parameter		**BSW Type**
FrNmGlobalFeatures		EcucParamConfContainerDef
BSW Description		
This container contains module features related to the FlexRay NM functionality.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
FrNm	FrNm/FrNmGlobalConfig/FrNmGlobalFeatures	
BSW Parameter		**BSW Type**
FrNmBusSynchronizationEnabled		EcucBooleanParamDef
BSW Description		
Pre-processor switch for enabling the bus synchronisation.		
M2 Template	**M2 Description**	
SystemTemplte	Enables bus synchronization support.	
M2 Parameter		
NetworkManagement::nmEcu.nmBusSynchronizationEnabled		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
FrNm	FrNm/FrNmGlobalConfig/FrNmGlobalFeatures	
BSW Parameter		**BSW Type**
FrNmComUserDataSupport		EcucBooleanParamDef
BSW Description		
Enable/disable the user data support.		
M2 Template	**M2 Description**	
SystemTemplate	Enable/disable the user data support.	
M2 Parameter		
FibexCore::CoreCommunication::NmPdu.nmDataInformation		
Mapping Rule		**Mapping Type**
If the nmDataInformation attribute is set to true for NmPdus that are transmitted by the regarded Ecu than this parameter must also be set to true.		full

BSW Module	BSW Context	
FrNm	FrNm/FrNmGlobalConfig/FrNmGlobalFeatures	
BSW Parameter		**BSW Type**
FrNmControlBitVectorEnabled		EcucBooleanParamDef
BSW Description		
Pre-processor switch for enabling control bit vector support. calculationFormula = If (FrNmNodeDetectionEnabled == False) then Equal(False) else Equal(False or True)		
M2 Template	**M2 Description**	
SystemTemplate	Enables control bit vector support.	
M2 Parameter		
NetworkManagement::FlexRayNmClusterCoupling.nmControlBitVectorEnabled		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
FrNm	FrNm/FrNmGlobalConfig/FrNmGlobalFeatures	
BSW Parameter		**BSW Type**
FrNmCoordinatorSyncSupport		EcucBooleanParamDef
BSW Description		
Enables/disables the coordinator synchronisation support.		
M2 Template	**M2 Description**	

M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**
FrNm	FrNm/FrNmGlobalConfig/FrNmGlobalFeatures
BSW Parameter	**BSW Type**
FrNmCycleCounterEmulation	EcucBooleanParamDef
BSW Description	
Pre-processor switch for enabling the cycle counter emulation.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**
FrNm	FrNm/FrNmGlobalConfig/FrNmGlobalFeatures
BSW Parameter	**BSW Type**
FrNmDualChannelPduEnable	EcucBooleanParamDef
BSW Description	
Pre-processor switch for enabling the support of dual channel transmission and reception of NM messages.	
M2 Template	**M2 Description**
SystemTemplate	Enables channel multiplicity support.
M2 Parameter	
NetworkManagement::NmEcu.nmMultipleChannelsEnabled	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	**BSW Context**
FrNm	FrNm/FrNmGlobalConfig/FrNmGlobalFeatures
BSW Parameter	**BSW Type**
FrNmHwVoteEnable	EcucBooleanParamDef
BSW Description	
Pre-processor switch for enabling the processing of FlexRay Hardware aggregated NM-Votes.	
M2 Template	**M2 Description**
SystemTemplate	Switch for enabling the processing of FlexRay Hardware aggregated NM-Votes.
M2 Parameter	
NetworkManagement::FlexRayNmEcu.hwVoteEnabled	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	**BSW Context**
FrNm	FrNm/FrNmGlobalConfig/FrNmGlobalFeatures
BSW Parameter	**BSW Type**

FrNmNodeDetectionEnabled	EcucBooleanParamDef
BSW Description	
Pre-processor switch for enabling node detection support. calculationFormula = If (FrNmPassiveModeEnabled == False) then Equal(NmNodeDetectionEnabled) else Equal(False)	
M2 Template	**M2 Description**
SystemTemplate	Enables the Request Repeat Message Request support. Only valid if nmNodeIdEnabled is set to true.
M2 Parameter	
NetworkManagement::NmEcu.nmNodeDetectionEnabled	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	**BSW Context**
FrNm	FrNm/FrNmGlobalConfig/FrNmGlobalFeatures
BSW Parameter	**BSW Type**
FrNmPassiveModeEnabled	EcucBooleanParamDef
BSW Description	
Pre-processor switch for enabling Passive Mode Configuration support. calculationFormula = Equal(NmPassiveModeEnabled)	
M2 Template	**M2 Description**
SystemTemplate	Enables support of the Passive Mode.
M2 Parameter	
NetworkManagement::nmEcu.nmPassiveModeEnabled	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	**BSW Context**
FrNm	FrNm/FrNmGlobalConfig/FrNmGlobalFeatures
BSW Parameter	**BSW Type**
FrNmPduRxIndicationEnabled	EcucBooleanParamDef
BSW Description	
Pre-processor switch for enabling PDU reception indication.	
M2 Template	**M2 Description**
SystemTemplate	Switch for enabling the PDU Rx Indication.
M2 Parameter	
NetworkManagement::nmEcu.nmPduRxIndicationEnabled	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	**BSW Context**
FrNm	FrNm/FrNmGlobalConfig/FrNmGlobalFeatures
BSW Parameter	**BSW Type**
FrNmPnEiraCalcEnabled	EcucBooleanParamDef
BSW Description	
Specifies if FrNm calculates the PN request information for internal an external requests. (EIRA) true: PN request are calculated false: PN request are not calculated	

M2 Template	M2 Description	
M2 Parameter		
Mapping Rule		**Mapping Type**

BSW Module	BSW Context		
FrNm	FrNm/FrNmGlobalConfig/FrNmGlobalFeatures		
BSW Parameter		**BSW Type**	
FrNmPnEiraRxNSduRef		EcucReferenceDef	
BSW Description			
Reference to a Pdu in the COM-Stack. Only one SduRef is required for FrNm because the EIRA is the aggregation over all FlexRay Channels.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	BSW Context		
FrNm	FrNm/FrNmGlobalConfig/FrNmGlobalFeatures		
BSW Parameter		**BSW Type**	
FrNmPnInfo		EcucParamConfContainerDef	
BSW Description			
PN information configuration			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	BSW Context		
FrNm	FrNm/FrNmGlobalConfig/FrNmGlobalFeatures/FrNmPnInfo		
BSW Parameter		**BSW Type**	
FrNmPnFilterMaskByte		EcucParamConfContainerDef	
BSW Description			
Filter mask byte configuration			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	BSW Context

FrNm	FrNm/FrNmGlobalConfig/FrNmGlobalFeatures/FrNmPnInfo/FrNmPnFilterMask Byte
BSW Parameter	**BSW Type**
FrNmPnFilterMaskByteIndex	EcucIntegerParamDef
BSW Description	
Index of the filter mask byte. Specifies the position within the filter mask byte array.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**
FrNm	FrNm/FrNmGlobalConfig/FrNmGlobalFeatures/FrNmPnInfo/FrNmPnFilterMask Byte
BSW Parameter	**BSW Type**
FrNmPnFilterMaskByteValue	EcucIntegerParamDef
BSW Description	
Parameter to configure the filter mask byte.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**
FrNm	FrNm/FrNmGlobalConfig/FrNmGlobalFeatures/FrNmPnInfo
BSW Parameter	**BSW Type**
FrNmPnInfoLength	EcucIntegerParamDef
BSW Description	
Specifies the length of the PN request information in the NM message.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**
FrNm	FrNm/FrNmGlobalConfig/FrNmGlobalFeatures/FrNmPnInfo
BSW Parameter	**BSW Type**
FrNmPnInfoOffset	EcucIntegerParamDef
BSW Description	
Specifies the offset of the PN request information in the NM message.	
M2 Template	**M2 Description**
M2 Parameter	

Mapping Rule	Mapping Type
	local

BSW Module	BSW Context	
FrNm	FrNm/FrNmGlobalConfig/FrNmGlobalFeatures	
BSW Parameter		**BSW Type**
FrNmPnResetTime		EcucFloatParamDef
BSW Description		
Specifies the runtime of the reset timer in seconds. This reset time is valid for the reset of PN requests in the EIRA and in the ERA. The value shall be the same for every channel. Thus it is a global config parameter.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**

BSW Module	BSW Context	
FrNm	FrNm/FrNmGlobalConfig/FrNmGlobalFeatures	
BSW Parameter		**BSW Type**
FrNmRemoteSleepIndicationEnabled		EcucBooleanParamDef
BSW Description		
Pre-processor switch for enabling remote sleep indication. calculationFormula = If (FrNmPassiveModeEnabled == True) then Equal(False) else Equal(False or True)		
M2 Template	**M2 Description**	
SystemTemplate	Switch for enabling the PDU Rx Indication.	
M2 Parameter		
NetworkManagement::NmEcu.nmRemoteSleepIndEnabled		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
FrNm	FrNm/FrNmGlobalConfig/FrNmGlobalFeatures	
BSW Parameter		**BSW Type**
FrNmRepeatMessageBitEnabled		EcucBooleanParamDef
BSW Description		
Pre-processor switch for enabling the repeat message bit support.		
M2 Template	**M2 Description**	
System Template	Switch for enabling the Repeat Message Bit Indication.	
M2 Parameter		
NetworkManagement::FlexRayNmEcu.nmRepeatMessageBitEnable		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context
FrNm	FrNm/FrNmGlobalConfig/FrNmGlobalFeatures

— AUTOSAR CONFIDENTIAL —

BSW Parameter		BSW Type
FrNmSourceNodeIdentifierEnabled		EcucBooleanParamDef
BSW Description		
Pre-processor switch for enabling SourceNodeIdentifier support.		
M2 Template	**M2 Description**	
SystemTemplate	Enables the source node identifier.	
M2 Parameter		
NetworkManagement::NmEcu.nmNodeIdEnabled		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
FrNm	FrNm/FrNmGlobalConfig/FrNmGlobalFeatures	
BSW Parameter		**BSW Type**
FrNmStateChangeIndicationEnabled		EcucBooleanParamDef
BSW Description		
Pre-processor switch for enabling state change indication.		
M2 Template	**M2 Description**	
SystemTemplate	Switch for enabling remote sleep indication support.	
M2 Parameter		
NetworkManagement::nmEcu.nmStateChangeIndEnabled		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
FrNm	FrNm/FrNmGlobalConfig/FrNmGlobalFeatures	
BSW Parameter		**BSW Type**
FrNmUserDataEnabled		EcucBooleanParamDef
BSW Description		
Pre-processor switch for enabling user data support.		
M2 Template	**M2 Description**	
SystemTemplate	Switch for enabling user data support.	
M2 Parameter		
NetworkManagement::NmEcu.nmUserDataEnabled		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
FrNm	FrNm/FrNmGlobalConfig/FrNmGlobalFeatures	
BSW Parameter		**BSW Type**
FrNmVotingNextToLastRepetitionCycleDisable		EcucBooleanParamDef
BSW Description		
Pre-processor switch for disabling vote changes in the last two repetition cycles before the Ready Sleep Counter expires.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
FrNm	FrNm/FrNmGlobalConfig	
BSW Parameter		**BSW Type**
FrNmGlobalProperties		EcucParamConfContainerDef
BSW Description		
This container contains module properties related to the FlexRay NM functionality.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
FrNm	FrNm/FrNmGlobalConfig/FrNmGlobalProperties	
BSW Parameter		**BSW Type**
FrNmDevErrorDetect		EcucBooleanParamDef
BSW Description		
Pre-processor switch for enabling development error detection		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
FrNm	FrNm/FrNmGlobalConfig/FrNmGlobalProperties	
BSW Parameter		**BSW Type**
FrNmMainAcrossFrCycle		EcucBooleanParamDef
BSW Description		
Parameter describing if the execution of FrNm_Main function crosses the FlexRay cycle boundary or not.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
FrNm	FrNm/FrNmGlobalConfig/FrNmGlobalProperties	
BSW Parameter		**BSW Type**
FrNmVersionInfoApi		EcucBooleanParamDef
BSW Description		
Pre-processor switch for enabling version info API support.		
M2 Template	**M2 Description**	
M2 Parameter		

— AUTOSAR CONFIDENTIAL —

Mapping Rule	Mapping Type
	local

D.9 FlexRayAutosarTp Mapping

BSW Module	BSW Context	
FrArTp	FrArTp	
BSW Parameter		**BSW Type**
FrArTpGeneral		EcucParamConfContainerDef
BSW Description		
This container contains the general configuration (parameters) of the FlexRay TP.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
FrArTp	FrArTp/FrArTpGeneral	
BSW Parameter		**BSW Type**
FrArTpChanNum		EcucIntegerParamDef
BSW Description		
Preprocessor switch for defining the number of concurrent channels the module supports. Up to 32 channels shall be definable here.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
FrArTp	FrArTp/FrArTpGeneral	
BSW Parameter		**BSW Type**
FrArTpDevErrorDetect		EcucBooleanParamDef
BSW Description		
Preprocessor switch for enabling development error detection.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
FrArTp	FrArTp/FrArTpGeneral	
BSW Parameter		**BSW Type**
FrArTpHaveAckRt		EcucBooleanParamDef

BSW Description		
Preprocessor switch for enabling the Acknowledgement and retry mechanisms.		
M2 Template	M2 Description	
M2 Parameter		
Mapping Rule		Mapping Type
		local

BSW Module	BSW Context	
FrArTp	FrArTp/FrArTpGeneral	
BSW Parameter		BSW Type
FrArTpHaveGrpSeg		EcucBooleanParamDef
BSW Description		
Preprocessor switch for enabling segmentation of 1:n messages.		
M2 Template	M2 Description	
M2 Parameter		
Mapping Rule		Mapping Type
		local

BSW Module	BSW Context	
FrArTp	FrArTp/FrArTpGeneral	
BSW Parameter		BSW Type
FrArTpHaveLm		EcucBooleanParamDef
BSW Description		
Preprocessor switch for enabling the mechanism for message longer than allowed by.		
M2 Template	M2 Description	
M2 Parameter		
Mapping Rule		Mapping Type
		local

BSW Module	BSW Context	
FrArTp	FrArTp/FrArTpGeneral	
BSW Parameter		BSW Type
FrArTpHaveTc		EcucBooleanParamDef
BSW Description		
Preprocessor switch for enabling Transmit Cancellation.		
M2 Template	M2 Description	
M2 Parameter		
Mapping Rule		Mapping Type
		local

BSW Module	BSW Context

FrArTp	FrArTp/FrArTpGeneral	
BSW Parameter		**BSW Type**
FrArTpMainFuncCycle		EcucFloatParamDef
BSW Description		
This parameter contains the calling period of the TPs Main Function. The parameter is specified in seconds.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
FrArTp	FrArTp/FrArTpGeneral	
BSW Parameter		**BSW Type**
FrArTpVersionInfoApi		EcucBooleanParamDef
BSW Description		
Preprocessor switch for enabling the Version info API.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
FrArTp	FrArTp	
BSW Parameter		**BSW Type**
FrArTpMultipleConfig		EcucParamConfContainerDef
BSW Description		
This container holds one or several multiple configuration sets.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
FrArTp	FrArTp/FrArTpMultipleConfig	
BSW Parameter		**BSW Type**
FrArTpChannel		EcucParamConfContainerDef
BSW Description		
This container contains the configuration (parameters) of one FlexRay TP channel.		
M2 Template	**M2 Description**	
SystemTemplate	A channel is a group of connections sharing several properties. The FlexRayArTp supports several channels. These channels can work concurrently, thus each of them requires its own state machine and management data structures and its own PDU-IDs.	

M2 Parameter	
TransportProtocols::FlexrayArTpChannel	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	BSW Context	
FrArTp	FrArTp/FrArTpMultipleConfig/FrArTpChannel	
BSW Parameter		**BSW Type**
FrArTpAckType		EcucEnumerationParamDef
BSW Description		
This parameter defines the type of acknowledgement which is used for the specific channel.		
M2 Template	**M2 Description**	
SystemTemplate	Type of Acknowledgement.	
M2 Parameter		
TransportProtocols::FlexrayArTpChannel.ackType		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
FrArTp	FrArTp/FrArTpMultipleConfig/FrArTpChannel	
BSW Parameter		**BSW Type**
FrArTpAdrType		EcucEnumerationParamDef
BSW Description		
This parameter states the addressing type this connection has. The meanings of the values are one byte and two byte.		
M2 Template	**M2 Description**	
SystemTemplate	Adressing Type of this connection: true: Two Bytes false: One Byte	
M2 Parameter		
TransportProtocols::FlexrayArTpChannel.extendedAddressing		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
FrArTp	FrArTp/FrArTpMultipleConfig/FrArTpChannel	
BSW Parameter		**BSW Type**
FrArTpChannelId		EcucIntegerParamDef
BSW Description		
The Id of the channel.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context
FrArTp	FrArTp/FrArTpMultipleConfig/FrArTpChannel

BSW Parameter		BSW Type	
FrArTpConNum		EcucIntegerParamDef	
BSW Description			
This parameter states the number of connections used in this channel. At least 256 shall be configurable here.			
M2 Template	**M2 Description**		
SystemTemplate	Group of connections that can be used in this channel.		
M2 Parameter			
TransportProtocols::FlexrayArTpChannel.tpConnection			
Mapping Rule			**Mapping Type**
Count aggregated TpConnections.			full

BSW Module	BSW Context		
FrArTp	FrArTp/FrArTpMultipleConfig/FrArTpChannel		
BSW Parameter		**BSW Type**	
FrArTpConnection		EcucParamConfContainerDef	
BSW Description			
This container contains the configuration (parameters) of one FlexRay TP connection. A connection can only belong to one channel.			
M2 Template	**M2 Description**		
SystemTemplate	Group of connections that can be used in this channel.		
M2 Parameter			
TransportProtocols::FlexrayArTpConnection			
Mapping Rule			**Mapping Type**
Create container for each existing FlexrayArTpConnection that is aggregated by FlexrayArTpChannel in the System description.			full

BSW Module	BSW Context		
FrArTp	FrArTp/FrArTpMultipleConfig/FrArTpChannel/FrArTpConnection		
BSW Parameter		**BSW Type**	
FrArTpConPduRef		EcucReferenceDef	
BSW Description			
Each value defines a PDU to be used for this connection. Thus each value is a PDU-ID given in FrArTpPdu and this array cannot be longer than the array FrArTpPdu. Please note: Only PDUs of the same size shall be used within a connection. Of course the PDU having the TxConfirmation configured has to be used by every connection.			
M2 Template	**M2 Description**		
SystemTemplate	Reference to an NPdu (Single Frame, First Frame or Consecutive Frame).		
M2 Parameter			
TransportProtocols::FlexrayArTpConnection.transmitPdu			
Mapping Rule			**Mapping Type**
1:1 mapping			full

BSW Module	BSW Context	
FrArTp	FrArTp/FrArTpMultipleConfig/FrArTpChannel/FrArTpConnection	
BSW Parameter		**BSW Type**
FrArTpLa		EcucIntegerParamDef
BSW Description		

— AUTOSAR CONFIDENTIAL —

This parameter defines the Local Address for the respective connection. When the local instance is the sender, this is the Source Address within the TP frame. When the local instance is the receiver, this is the Target Address within the TP frame. Note that in case of 1 byte addressing only the values from 0x0000 - 0x00FF are valid.		
M2 Template	**M2 Description**	
SystemTemplate	The source of the TP connection.	
M2 Parameter		
TransportProtocols::FlexrayArTpConnection.source		
Mapping Rule		**Mapping Type**
LocalAddress can be derived from the TpNode that is referenced by the Flex RayTpConnection as source.		full

BSW Module	**BSW Context**	
FrArTp	FrArTp/FrArTpMultipleConfig/FrArTpChannel/FrArTpConnection	
BSW Parameter		**BSW Type**
FrArTpMultRec		EcucBooleanParamDef
BSW Description		
This parameter defines, whether this connection is an 1:1 ('false') or an 1:n ('true') connection. Of course, if the channel to which the connection is configured has retry or acknowledgement enabled, no retry or acknowledgement will occur in case the connection is an 1:n connection.		
M2 Template	**M2 Description**	
SystemTemplate	TP address for 1:n connections.	
M2 Parameter		
TransportProtocols::FlexrayArTpConnection.multicast		
Mapping Rule		**Mapping Type**
If multicast is used set this attribute to true.		full

BSW Module	**BSW Context**	
FrArTp	FrArTp/FrArTpMultipleConfig/FrArTpChannel/FrArTpConnection	
BSW Parameter		**BSW Type**
FrArTpRa		EcucIntegerParamDef
BSW Description		
This parameter defines the Remote Address for the respective connection. When the local instance is the sender, this is the Target Address within the TP frame. When the local instance is the receiver, this is the Source Address within the TP frame. Note that in case of 1 byte addressing only the values from 0x0000 - 0x00FF are valid.		
M2 Template	**M2 Description**	
SystemTemplate	The target of the TP connection.	
M2 Parameter		
TransportProtocols::FlexrayArTpConnection.sourceTransportProtocols::FlexrayArTpConnection.target		
Mapping Rule		**Mapping Type**
RemoteAddress can be derived from the TpNode that is referenced by the Flex RayTpConnection as target.		full

BSW Module	**BSW Context**	
FrArTp	FrArTp/FrArTpMultipleConfig/FrArTpChannel/FrArTpConnection	
BSW Parameter		**BSW Type**
FrArTpRxSdu		EcucParamConfContainerDef
BSW Description		
Describes the Rx SDU		

M2 Template	M2 Description	
SystemTemplate	Reference to the IPdu that is segmented by the Transport Protocol.	
M2 Parameter		
TransportProtocols::FlexrayArTpConnection.directTpSdu		
Mapping Rule		**Mapping Type**
Create container for every IPdu that is received by the FrArTp and the regarded Ecu.		full

BSW Module	BSW Context	
FrArTp	FrArTp/FrArTpMultipleConfig/FrArTpChannel/FrArTpConnection/FrArTpRxSdu	
BSW Parameter		**BSW Type**
FrArTpRxSduRef		EcucReferenceDef
BSW Description		
Reference to a PDU in the global PDU structure.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
FrArTp	FrArTp/FrArTpMultipleConfig/FrArTpChannel/FrArTpConnection/FrArTpRxSdu	
BSW Parameter		**BSW Type**
FrArTpSduRxId		EcucIntegerParamDef
BSW Description		
This is a unique identifier for a received message. This Id is used in the CancelReceive API call. ImplementationType: PduIdType		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
FrArTp	FrArTp/FrArTpMultipleConfig/FrArTpChannel/FrArTpConnection	
BSW Parameter		**BSW Type**
FrArTpTxSdu		EcucParamConfContainerDef
BSW Description		
Describes the Tx SDU		
M2 Template	**M2 Description**	
SystemTemplate	Reference to the IPdu that is segmented by the Transport Protocol.	
M2 Parameter		
Mapping Rule		**Mapping Type**
Create container for every IPdu that is transmitted by the FrArTp and the regarded Ecu.		full

BSW Module	BSW Context	
FrArTp	FrArTp/FrArTpMultipleConfig/FrArTpChannel/FrArTpConnection/FrArTpTxSdu	
BSW Parameter		**BSW Type**
FrArTpSduTxId		EcucIntegerParamDef
BSW Description		
This is a unique identifier for a received or a to be transmitted message. With this (and by means of e.g. a lookup table) the PDU Router can route the message appropriately without dealing with the particularities of the Transport Layer. This parameter can also be seen as the identifier of a connection. ImplementationType: PduIdType		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
FrArTp	FrArTp/FrArTpMultipleConfig/FrArTpChannel/FrArTpConnection/FrArTpTxSdu	
BSW Parameter		**BSW Type**
FrArTpTxSduRef		EcucReferenceDef
BSW Description		
Reference to a PDU in the global PDU structure.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
FrArTp	FrArTp/FrArTpMultipleConfig/FrArTpChannel	
BSW Parameter		**BSW Type**
FrArTpGrpSeg		EcucBooleanParamDef
BSW Description		
Here can be specified, whether segmentation within a 1:n connection is allowed or not.		
M2 Template	**M2 Description**	
SystemTemplate	This attribute defines whether segmentation within a 1:n connection is allowed or not.	
M2 Parameter		
TransportProtocols::FlexrayArTpChannel.multicastSegmentation		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
FrArTp	FrArTp/FrArTpMultipleConfig/FrArTpChannel	
BSW Parameter		**BSW Type**
FrArTpLm		EcucEnumerationParamDef
BSW Description		

This specifies the maximum message length for the particular channel.		
M2 Template	**M2 Description**	
SystemTemplate	This specifies the maximum message length for the particular channel.	
M2 Parameter		
TransportProtocols::FlexrayArTpChannel.maximumMessageLength		
Mapping Rule		**Mapping Type**
		full

BSW Module	**BSW Context**	
FrArTp	FrArTp/FrArTpMultipleConfig/FrArTpChannel	
BSW Parameter		**BSW Type**
FrArTpMaxAr		EcucIntegerParamDef
BSW Description		
This parameter defines the maximum number of trying to send a frame when a TIMEOUT AR occurs (depending on whether retry is configured).		
M2 Template	**M2 Description**	
SystemTemplate	This attribute defines the maximum number of trying to send a frame when a TIMEOUT AR occurs (depending on whether retry is configured).	
M2 Parameter		
TransportProtocols::FlexrayArTpChannel.maxAr		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
FrArTp	FrArTp/FrArTpMultipleConfig/FrArTpChannel	
BSW Parameter		**BSW Type**
FrArTpMaxAs		EcucIntegerParamDef
BSW Description		
This parameter defines the maximum number of trying t osend a frame when a TIMEOUT AS occurs (depending on whether retry is configured)		
M2 Template	**M2 Description**	
SystemTemplate	This attribute defines the maximum number of trying to send a frame when a TIMEOUT AS occurs (depending on whether retry is configured).	
M2 Parameter		
TransportProtocols::FlexrayArTpChannel.maxAs		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
FrArTp	FrArTp/FrArTpMultipleConfig/FrArTpChannel	
BSW Parameter		**BSW Type**
FrArTpMaxBs		EcucIntegerParamDef
BSW Description		
This parameter defines number of consecutive CFs between two FCs (block size). Valid values are 1 .. 16 when retry is activated, and 0 .. 255 otherwise.		
M2 Template	**M2 Description**	
SystemTemplate	This attribute is only relevant when having retry activated. It limits the maximal block size the FrTp can choose in order to limit the amount of Tx buffer that will be requested at the sender side in a segmented transfer.	
M2 Parameter		

TransportProtocols::FlexrayArTpChannel.maxBs	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	**BSW Context**	
FrArTp	FrArTp/FrArTpMultipleConfig/FrArTpChannel	
BSW Parameter		**BSW Type**
FrArTpMaxBufReq		EcucIntegerParamDef
BSW Description		
This parameter defines the maximum number of times the FrArTp should send a wait frame FC(WT). It is also used to limit the number of retries for PduR_FrArTpCopyTxData and PduR_FrArTpCopyRxData when no timer is active.		
M2 Template	**M2 Description**	
SystemTemplate	This attribute defines the maximum number of trying to get a buffer (Transmit / Receive), depending of the return value of PduR_FrTpProvideTxBuffer / PduR_FrTpProvideRxBuffer and on whether retry is configured.	
M2 Parameter		
TransportProtocols::FlexrayArTpChannel.maxBufferRequest		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
FrArTp	FrArTp/FrArTpMultipleConfig/FrArTpChannel	
BSW Parameter		**BSW Type**
FrArTpMaxFrIf		EcucIntegerParamDef
BSW Description		
This parameter defines the maximum number of trying to send a frame when the FrIf returns an error.		
M2 Template	**M2 Description**	
SystemTemplate	This attribute defines the maximum number of trying to send a frame when the FrIf returns an error.	
M2 Parameter		
TransportProtocols::FlexrayArTpChannel.maxFrIf		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
FrArTp	FrArTp/FrArTpMultipleConfig/FrArTpChannel	
BSW Parameter		**BSW Type**
FrArTpMaxRn		EcucIntegerParamDef
BSW Description		
This parameter defines the maximum number of retries (if retry is configured for the particular channel).		
M2 Template	**M2 Description**	
SystemTemplate	This attribute defines the maximum number of retries (if retry is configured for the particular channel).	
M2 Parameter		
TransportProtocols::FlexrayArTpChannel.maxRetries		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context		
FrArTp	FrArTp/FrArTpMultipleConfig/FrArTpChannel		
BSW Parameter		**BSW Type**	
FrArTpPdu		EcucParamConfContainerDef	
BSW Description			
Container to hold the PDU parameters. ImplementationType: PduInfoType			
M2 Template	**M2 Description**		
SystemTemplate	A FlexRayTpChannel contains a pool of NPdus.		
M2 Parameter			
TransportProtocols::FlexrayArTpChannel.pduPool			
Mapping Rule			**Mapping Type**
Create container for each NPdu that is aggregated by FlexrayArTpChannel in the System description.			full

BSW Module	BSW Context		
FrArTp	FrArTp/FrArTpMultipleConfig/FrArTpChannel/FrArTpPdu		
BSW Parameter		**BSW Type**	
FrArTpPduDirection		EcucEnumerationParamDef	
BSW Description			
This parameter defines the direction of the PDU.			
M2 Template	**M2 Description**		
SystemTemplate	communication Direction of the Connector Port (input or output Port).		
M2 Parameter			
CoreCommunication::PduTriggering.iPduPort			
Mapping Rule			**Mapping Type**
The direction of the Npdu can be derived from the triggering elements that contain references to IN- and OUT-Ports.			full

BSW Module	BSW Context		
FrArTp	FrArTp/FrArTpMultipleConfig/FrArTpChannel/FrArTpPdu		
BSW Parameter		**BSW Type**	
FrArTpPduId		EcucIntegerParamDef	
BSW Description			
This is the identifier of the FlexRay Interface PDUs (Fr N-PDU, Fr L-SDU) in which the Transport Layer Frames of this channel should be transmitted. ImplementationType: PduIdType			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	BSW Context		
FrArTp	FrArTp/FrArTpMultipleConfig/FrArTpChannel/FrArTpPdu		
BSW Parameter		**BSW Type**	
FrArTpPduRef		EcucReferenceDef	

BSW Description		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
FrArTp	FrArTp/FrArTpMultipleConfig/FrArTpChannel	
BSW Parameter		**BSW Type**
FrArTpPduFc		EcucParamConfContainerDef
BSW Description		
This is the identifier of the FlexRay Interface PDUs (Fr N-PDU, Fr L-SDU) in which the Transport Layer Flow Control and Acknowledgement Frames of this channel should be transmitted. ImplementationType: PduInfoType		
M2 Template	**M2 Description**	
SystemTemplate	Reference to the Flow Control NPdu.	
M2 Parameter		
TransportProtocols::FlexrayArTpChannel.flowControlPdu		
Mapping Rule		**Mapping Type**
Create container for every FlowControlPdu that is referenced by the FlexrayArTpChannel.		full

BSW Module	**BSW Context**	
FrArTp	FrArTp/FrArTpMultipleConfig/FrArTpChannel/FrArTpPduFc	
BSW Parameter		**BSW Type**
FrArTpPduFcDirection		EcucEnumerationParamDef
BSW Description		
This parameter defines the direction of the PDU.		
M2 Template	**M2 Description**	
SystemTemplate	communication Direction of the Connector Port (input or output Port).	
M2 Parameter		
CoreCommunication::PduTriggering.iPduPort		
Mapping Rule		**Mapping Type**
The direction of the Npdu can be derived from the triggering elements that contain references to IN- and OUT-Ports.		full

BSW Module	**BSW Context**	
FrArTp	FrArTp/FrArTpMultipleConfig/FrArTpChannel/FrArTpPduFc	
BSW Parameter		**BSW Type**
FrArTpPduFcId		EcucIntegerParamDef
BSW Description		
This is the identifier of the FlexRay Interface PDUs (Fr N-PDU, Fr L-SDU) in which the Transport Layer Flow Control and Acknowledgement Frames of this channel should be transmitted.		
M2 Template	**M2 Description**	
M2 Parameter		

Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**	
FrArTp	FrArTp/FrArTpMultipleConfig/FrArTpChannel/FrArTpPduFc	
BSW Parameter		**BSW Type**
FrArTpPduFcRef		EcucReferenceDef
BSW Description		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
FrArTp	FrArTp/FrArTpMultipleConfig/FrArTpChannel	
BSW Parameter		**BSW Type**
FrArTpStMin		EcucFloatParamDef
BSW Description		
This parameter defines the minimum amount of time between two succeeding CFs in seconds. Valid values are 0, 100μs, 200μs .. 900μs, 1ms, 2ms .. 127ms. The value can be changed at runtime using the FrArTp_ChangeParameter interface.		
M2 Template	**M2 Description**	
SystemTemplate	This attribute defines the minimum amount of time (separation Time) between two succeeding CFs. Specified in seconds.	
M2 Parameter		
TransportProtocols::FlexrayArTpChannel.minimumSeparationTime		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
FrArTp	FrArTp/FrArTpMultipleConfig/FrArTpChannel	
BSW Parameter		**BSW Type**
FrArTpTc		EcucBooleanParamDef
BSW Description		
With this switch Transmit Cancellation can be turned on or off for this channel.		
M2 Template	**M2 Description**	
SystemTemplate	This attribute states whether Transmit Cancellation is supported on this channel.	
M2 Parameter		
TransportProtocols::FlexrayArTpChannel.transmitCancellation		
Mapping Rule		**Mapping Type**
1:1 mapping		local

BSW Module	**BSW Context**	
FrArTp	FrArTp/FrArTpMultipleConfig/FrArTpChannel	
BSW Parameter		**BSW Type**

FrArTpTimeBr	EcucFloatParamDef
BSW Description	
This parameter defines the time in seconds between receiving the last CF of a block or an FF-x (or SF-x) and sending out an FC or AF. It is obvious that FRARTP_TIME_BR + FRARTP_TIMEOUT_AR < FRARTP_TIMEOUT_BS must hold (because the transmission duration on the bus has also to be considered). This parameter is defined in ISO 15765-2. It is contained in the configuration as a performance requirement.	
M2 Template	**M2 Description**
SystemTemplate	This attribute defines the time in seconds between receiving the last CF of a block or an FF-x (or SF-x) and sending out an FC or AF.
M2 Parameter	
TransportProtocols::FlexrayArTpChannel.timeBr	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	**BSW Context**
FrArTp	FrArTp/FrArTpMultipleConfig/FrArTpChannel
BSW Parameter	**BSW Type**
FrArTpTimeBuffer	EcucFloatParamDef
BSW Description	
This parameter defines the time in seconds of waiting for the next try (if retry is activated) to get a Tx or Rx buffer.	
M2 Template	**M2 Description**
SystemTemplate	This attribute defines the time in seconds of waiting for the next try (if retry is activated) to get a Tx or Rx buffer.
M2 Parameter	
TransportProtocols::FlexrayArTpChannel.timeBuffer	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	**BSW Context**
FrArTp	FrArTp/FrArTpMultipleConfig/FrArTpChannel
BSW Parameter	**BSW Type**
FrArTpTimeCs	EcucFloatParamDef
BSW Description	
This parameter defines the time in seconds between the sending of two consecutive CFs or between reception of an FC or AF and sending of the next CF . It is obvious that FRARTP_TIME_CS + FRARTP_TIMEOUT_AS < FRARTP_TIMEOUT_CR must hold (because the transmission duration on the bus has also to be considered). This parameter is defined in ISO 15765-2. It is contained in the configuration as a performance requirement.	
M2 Template	**M2 Description**
SystemTemplate	Defines the time in seconds between the sending of two consecutive frames or between a consecutive frame and a flow control or between reception of an flow control or Acknowledgement Frame and sending of the next consecutive frame or a flow control.
M2 Parameter	
TransportProtocols::FlexrayArTpChannel.timeCs	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	BSW Context	
FrArTp	FrArTp/FrArTpMultipleConfig/FrArTpChannel	
BSW Parameter		**BSW Type**
FrArTpTimeFrIf		EcucFloatParamDef
BSW Description		
This parameter defines the time in seconds of waiting for the next try (if retry is activated) to send via FrIf_Transmit.		
M2 Template	**M2 Description**	
SystemTemplate	This attribute defines the time in seconds of waiting for the next try (if retry is activated) to send via FrIf_Transmit. Specified in seconds.	
M2 Parameter		
TransportProtocols::FlexrayArTpChannel.timeFrIf		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
FrArTp	FrArTp/FrArTpMultipleConfig/FrArTpChannel	
BSW Parameter		**BSW Type**
FrArTpTimeoutAr		EcucFloatParamDef
BSW Description		
This parameter states the timeout in seconds between the PDU transmit request of the Transport Layer to the FlexRay Interface and the corresponding confirmation of the FlexRay Interface on the receiver side (for FC or AF).		
M2 Template	**M2 Description**	
SystemTemplate	This attribute states the timeout in seconds between the PDU transmit request of the Transport Layer to the FlexRay Interface and the corresponding confirmation of the FlexRay Interface on the receiver side (for FC or AF).	
M2 Parameter		
TransportProtocols::FlexrayArTpChannel.timeoutAr		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
FrArTp	FrArTp/FrArTpMultipleConfig/FrArTpChannel	
BSW Parameter		**BSW Type**
FrArTpTimeoutAs		EcucFloatParamDef
BSW Description		
This parameter states the timeout in seconds between the PDU transmit request for the first PDU of the group used in the current connection of the Transport Layer to the FlexRay Interface and the corresponding confirmation of the FlexRay Interface (when having sent the last PDU of the group used in this connection) on the sender side (SF-x, FF-x, CF).		
M2 Template	**M2 Description**	
SystemTemplate	This attribute states the timeout in seconds between the PDU transmit request for the first PDU of the group used in the current connection of the Transport Layer to the FlexRay Interface and the corresponding confirmation of the FlexRay Interface.	
M2 Parameter		
TransportProtocols::FlexrayArTpChannel.timeoutAs		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
FrArTp	FrArTp/FrArTpMultipleConfig/FrArTpChannel	
BSW Parameter		**BSW Type**
FrArTpTimeoutBs		EcucFloatParamDef
BSW Description		
This parameter defines the timeout in seconds for waiting for an FC or AF on the sender side in a 1:1 connection.		
M2 Template	**M2 Description**	
SystemTemplate	This attribute defines the timeout in seconds for waiting for an FC or AF on the sender side in a 1:1 connection.	
M2 Parameter		
TransportProtocols::FlexrayArTpChannel.timeoutBs		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
FrArTp	FrArTp/FrArTpMultipleConfig/FrArTpChannel	
BSW Parameter		**BSW Type**
FrArTpTimeoutCr		EcucFloatParamDef
BSW Description		
This parameter defines the timeout value in seconds for waiting for a CF or FF-x (in case of retry) after receiving the last CF or after sending an FC or AF on the receiver side.		
M2 Template	**M2 Description**	
SystemTemplate	This attribute defines the timeout value in seconds for waiting for a CF or FF-x (in case of retry) after receiving the last CF or after sending an FC or AF on the receiver side.	
M2 Parameter		
TransportProtocols::FlexrayArTpChannel.timeoutCr		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
FrArTp	FrArTp/FrArTpMultipleConfig/FrArTpChannel	
BSW Parameter		**BSW Type**
FrArTpUsePduFc		EcucBooleanParamDef
BSW Description		
This switch defines, whether within this channel the dedicated FC/ACK PDU (FrArTpPduFc) shall be used or not. If this is not used FC / ACK frames are sent using the normal IDs, otherwise only FrArTpPduFc shall be used for sending / receiving FC / ACK frames.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

D.10 FlexRayIsoTp Mapping

BSW Module	BSW Context
FrTp	FrTp

BSW Parameter		BSW Type
FrTpGeneral		EcucParamConfContainerDef
BSW Description		
This container contains the general configuration parameters of the FlexRay Transport Protocol module.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**

BSW Module	BSW Context		
FrTp	FrTp/FrTpGeneral		
BSW Parameter		**BSW Type**	
FrTpAckRt		EcucBooleanParamDef	
BSW Description			
Preprocessor switch for enabling the Acknowledgement and retry mechanisms. True: Acknowledge and Retry is enabled False: Acknowledge and Retry is disabled			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	BSW Context		
FrTp	FrTp/FrTpGeneral		
BSW Parameter		**BSW Type**	
FrTpChanNum		EcucIntegerParamDef	
BSW Description			
Preprocessor switch for defining the number of concurrent channels the module supports. Up to 32 channels shall be definable here.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	BSW Context	
FrTp	FrTp/FrTpGeneral	
BSW Parameter		**BSW Type**
FrTpChangeParamApi		EcucBooleanParamDef
BSW Description		
Preprocessor switch for enabling the API to change FrTp communication parameters. True: Version Info API is enabled False: Version Info API is disabled		

M2 Template	M2 Description	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context		
FrTp	FrTp/FrTpGeneral		
BSW Parameter		**BSW Type**	
FrTpDevErrorDetect		EcucBooleanParamDef	
BSW Description			
Preprocessor switch for enabling development error detection. True: Development Error Detection is enabled False: Development Error Detection is disabled			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	BSW Context		
FrTp	FrTp/FrTpGeneral		
BSW Parameter		**BSW Type**	
FrTpFullDuplexEnable		EcucBooleanParamDef	
BSW Description			
Preprocessor switch for enabling full duplex mechanisms for all channels. True: Full duplex is enabled False: Fullduplex is disabled (Half duplex is enabled)			
M2 Template	**M2 Description**		
SystemTemplate	The full duplex mechanisms is enabled if this attribute is set to true. Otherwise half duplex is enabled.		
M2 Parameter			
TransportProtocols::TpConnectionControl.fullDuplexEnabled			
Mapping Rule			**Mapping Type**
1:1 mapping			local

BSW Module	BSW Context	
FrTp	FrTp/FrTpGeneral	
BSW Parameter		**BSW Type**
FrTpMainFuncCycle		EcucFloatParamDef
BSW Description		
This parameter contains the calling period of the TPs Main Function. The parameter is specified in seconds.		
M2 Template	**M2 Description**	
SystemTemplate	The period between successive calls to the Main Function of the ASR TP. Specified in seconds.	
M2 Parameter		

TransportProtocols::TpEcu.cycleTimeMainFunction	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	**BSW Context**	
FrTp	FrTp/FrTpGeneral	
BSW Parameter		**BSW Type**
FrTpTransmitCancellation		EcucBooleanParamDef
BSW Description		
Preprocessor switch for enabling Transmit Cancellation. True: Transmit Cancellation is enabled False: Transmit Cancellation is disabled		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
FrTp	FrTp/FrTpGeneral	
BSW Parameter		**BSW Type**
FrTpUnknownMsgLength		EcucBooleanParamDef
BSW Description		
Preprocessor switch to support data transfer with unknown message length. True: Transmission with unknown message length is enabled False: Transmission with unknown message length is disabled		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
FrTp	FrTp/FrTpGeneral	
BSW Parameter		**BSW Type**
FrTpVersionInfoApi		EcucBooleanParamDef
BSW Description		
Preprocessor switch for enabling the Version info API. True: Version Info API is enabled False: Version Info API is disabled		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**		
FrTp	FrTp		
BSW Parameter		**BSW Type**	
FrTpMultipleConfig		EcucParamConfContainerDef	
BSW Description			
This container holds one or several multiple configuration sets.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	**BSW Context**		
FrTp	FrTp/FrTpMultipleConfig		
BSW Parameter		**BSW Type**	
FrTpConnection		EcucParamConfContainerDef	
BSW Description			
This container contains the connection specific parameters to transfer N-PDUs via FlexRay TP.			
M2 Template	**M2 Description**		
SystemTemplate	A connection identifies the sender and the receiver of this particular communication. The FlexRayTp module routes a Pdu through this connection.		
M2 Parameter			
TransportProtocols::FlexRayTpConnection			
Mapping Rule			**Mapping Type**
Create container for each FlexRayTpConnection that is described in the ECU Extract.			full

BSW Module	**BSW Context**		
FrTp	FrTp/FrTpMultipleConfig/FrTpConnection		
BSW Parameter		**BSW Type**	
FrTpBandwidthLimitation		EcucBooleanParamDef	
BSW Description			
This parameter indicates wheather the connection requires a bandwidth limitation or not. If FrTpBandwidthLimitation=True the sender shall send a StartFrame always on the first PDU of a PDU-Pool.			
M2 Template	**M2 Description**		
SystemTemplate	Specifies whether the connection requires a bandwidth		
M2 Parameter			
TransportProtocols::FlexRayTpConnection.bandwidthLimitation			
Mapping Rule			**Mapping Type**
1:1 mapping			full

BSW Module	**BSW Context**	
FrTp	FrTp/FrTpMultipleConfig/FrTpConnection	
BSW Parameter		**BSW Type**
FrTpConCtrlRef		EcucReferenceDef
BSW Description		
FrTpConnectionControlReference: This parameter defines a reference to a connection control container.		

<table>
<tr><th>M2 Template</th><th colspan="2">M2 Description</th></tr>
<tr><td>SystemTemplate</td><td colspan="2">Reference to the connection control.</td></tr>
<tr><th colspan="3">M2 Parameter</th></tr>
<tr><td colspan="3">TransportProtocols::FlexRayTpConnection.tpConnectionControl</td></tr>
<tr><th colspan="2">Mapping Rule</th><th>Mapping Type</th></tr>
<tr><td colspan="2">1:1 mapping</td><td>full</td></tr>
</table>

<table>
<tr><th>BSW Module</th><th colspan="3">BSW Context</th></tr>
<tr><td>FrTp</td><td colspan="3">FrTp/FrTpMultipleConfig/FrTpConnection</td></tr>
<tr><th colspan="2">BSW Parameter</th><th colspan="2">BSW Type</th></tr>
<tr><td colspan="2">FrTpLa</td><td colspan="2">EcucIntegerParamDef</td></tr>
<tr><th colspan="4">BSW Description</th></tr>
<tr><td colspan="4">This parameter defines the Local Address for the respective connection. When the local instance is the sender, this is the Source Address within the TP frame. When the local instance is the receiver, this is the Target Address within the TP frame.</td></tr>
<tr><th>M2 Template</th><th colspan="3">M2 Description</th></tr>
<tr><td>SystemTemplate</td><td colspan="3">The source of the TP connection.</td></tr>
<tr><th colspan="4">M2 Parameter</th></tr>
<tr><td colspan="4">TransportProtocols::FlexRayTpConnection.transmitter</td></tr>
<tr><th colspan="3">Mapping Rule</th><th>Mapping Type</th></tr>
<tr><td colspan="3">FlexRayTpConnection contains a reference to a TpNode. TpNode references the TpAddress element.</td><td>full</td></tr>
</table>

<table>
<tr><th>BSW Module</th><th colspan="3">BSW Context</th></tr>
<tr><td>FrTp</td><td colspan="3">FrTp/FrTpMultipleConfig/FrTpConnection</td></tr>
<tr><th colspan="2">BSW Parameter</th><th colspan="2">BSW Type</th></tr>
<tr><td colspan="2">FrTpMultipleReceiverCon</td><td colspan="2">EcucBooleanParamDef</td></tr>
<tr><th colspan="4">BSW Description</th></tr>
<tr><td colspan="4">This parameter defines, whether this connection is an 1:1 ('false') or an 1:n ('true') connection. If data segmentation is required this parameter is used to check whether segmentation is possible or not. If the connection is 1:n segmentation is not possible and an error will occur.</td></tr>
<tr><th>M2 Template</th><th colspan="3">M2 Description</th></tr>
<tr><td>SystemTemplate</td><td colspan="3">This parameter defines, whether this connection is an 1:1 ('false') or an 1:n ('true') connection.</td></tr>
<tr><th colspan="4">M2 Parameter</th></tr>
<tr><td colspan="4">TransportProtocols::FlexRayTpConnection.multicast</td></tr>
<tr><th colspan="3">Mapping Rule</th><th>Mapping Type</th></tr>
<tr><td colspan="3">If FlexRayTpConnection contains a mulicast reference to TpAddress than set this parameter to true</td><td>full</td></tr>
</table>

<table>
<tr><th>BSW Module</th><th colspan="2">BSW Context</th></tr>
<tr><td>FrTp</td><td colspan="2">FrTp/FrTpMultipleConfig/FrTpConnection</td></tr>
<tr><th colspan="2">BSW Parameter</th><th>BSW Type</th></tr>
<tr><td colspan="2">FrTpRa</td><td>EcucIntegerParamDef</td></tr>
<tr><th colspan="3">BSW Description</th></tr>
<tr><td colspan="3">This parameter defines the Remote Address for the respective connection. When the local instance is the sender, this is the Target Address within the TP frame. When the local instance is the receiver, this is the Source Address within the TP frame.</td></tr>
<tr><th>M2 Template</th><th colspan="2">M2 Description</th></tr>
<tr><td>SystemTemplate</td><td colspan="2">The target of the TP connection.</td></tr>
<tr><th colspan="3">M2 Parameter</th></tr>
</table>

TransportProtocols::FlexRayTpConnection.receiver	
Mapping Rule	**Mapping Type**
FlexRayTpConnection contains a reference to a TpNode. TpNode references the TpAddress element.	full

BSW Module	**BSW Context**	
FrTp	FrTp/FrTpMultipleConfig/FrTpConnection	
BSW Parameter		**BSW Type**
FrTpRxPduPoolRef		EcucReferenceDef
BSW Description		
This parameter defines a reference to a RxPduPool.		
M2 Template	**M2 Description**	
SystemTemplate	Reference to a pool of NPdus.	
M2 Parameter		
TransportProtocols::FlexRayTpConnection.rxPduPool		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
FrTp	FrTp/FrTpMultipleConfig/FrTpConnection	
BSW Parameter		**BSW Type**
FrTpRxSdu		EcucParamConfContainerDef
BSW Description		
This parameter defines the Rx Service Data Unit Identifier (Sdu Id) which uniquely identifies a data transfer (inter-module communication) between FrTp and PDUR.		
M2 Template	**M2 Description**	
SystemTemplate	Reference to the IPdu that is segmented by the Transport Protocol.	
M2 Parameter		
TransportProtocols::FlexRayTpConnection.directTpSdu		
Mapping Rule		**Mapping Type**
Create container if an Rx Pdu is referenced by the FlexRayTpConnection		full

BSW Module	**BSW Context**	
FrTp	FrTp/FrTpMultipleConfig/FrTpConnection/FrTpRxSdu	
BSW Parameter		**BSW Type**
FrTpRxSduId		EcucIntegerParamDef
BSW Description		
This unique identifier is used for change parameter request or receive cancellation from PduR to FrTp. ImplementationType: PduIdType		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**

FrTp	FrTp/FrTpMultipleConfig/FrTpConnection/FrTpRxSdu	
BSW Parameter		**BSW Type**
FrTpRxSduRef		EcucReferenceDef
BSW Description		
Reference to a PDU in the global PDU structure.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
FrTp	FrTp/FrTpMultipleConfig/FrTpConnection	
BSW Parameter		**BSW Type**
FrTpTxPduPoolRef		EcucReferenceDef
BSW Description		
This parameter defines a reference to a TxPduPool.		
M2 Template	**M2 Description**	
SystemTemplate	Reference to a pool of NPdus.	
M2 Parameter		
TransportProtocols::FlexRayTpConnection.txPduPool		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
FrTp	FrTp/FrTpMultipleConfig/FrTpConnection	
BSW Parameter		**BSW Type**
FrTpTxSdu		EcucParamConfContainerDef
BSW Description		
This parameter defines the Tx Service Data Unit Identifier (Sdu Id) which uniquely identifies a data transfer (inter-module communication) between FrTp and PDUR.		
M2 Template	**M2 Description**	
SystemTemplate	Reference to the IPdu that is segmented by the Transport Protocol.	
M2 Parameter		
TransportProtocols::FlexRayTpConnection.directTpSdu		
Mapping Rule		**Mapping Type**
Create container if an Tx Pdu is referenced by the FlexRayTpConnection		full

BSW Module	**BSW Context**	
FrTp	FrTp/FrTpMultipleConfig/FrTpConnection/FrTpTxSdu	
BSW Parameter		**BSW Type**
FrTpTxSduId		EcucIntegerParamDef
BSW Description		
This is a unique identifier for a to be transmitted message from the PduR to the FrTp. ImplementationType: PduIdType		
M2 Template	**M2 Description**	
M2 Parameter		

Mapping Rule	**Mapping Type**
	local

BSW Module	BSW Context	
FrTp	FrTp/FrTpMultipleConfig/FrTpConnection/FrTpTxSdu	
BSW Parameter		**BSW Type**
FrTpTxSduRef		EcucReferenceDef
BSW Description		
Reference to a PDU in the global PDU structure.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
FrTp	FrTp/FrTpMultipleConfig	
BSW Parameter		**BSW Type**
FrTpConnectionControl		EcucParamConfContainerDef
BSW Description		
This container contains the configuration parameters to control a FlexRay TP connection.		
M2 Template	**M2 Description**	
SystemTemplate	Configuration parameters to control a FlexRay TP connection.	
M2 Parameter		
TransportProtocols::FlexRayTpConnectionControl		
Mapping Rule		**Mapping Type**
Create container for each FlexRayTpConnectionControl that is described in the ECU Extract.		full

BSW Module	BSW Context	
FrTp	FrTp/FrTpMultipleConfig/FrTpConnectionControl	
BSW Parameter		**BSW Type**
FrTpAckType		EcucEnumerationParamDef
BSW Description		
This parameter defines the type of acknowledgement which is used for the specific channel.		
M2 Template	**M2 Description**	
SystemTemplate	This parameter defines the type of acknowledgement which is used for the specific channel.	
M2 Parameter		
TransportProtocols::FlexRayTpConnectionControl.ackType		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
FrTp	FrTp/FrTpMultipleConfig/FrTpConnectionControl	
BSW Parameter		**BSW Type**
FrTpMaxAr		EcucIntegerParamDef

BSW Description		
This parameter defines the maximum number of trying to send a frame when a TIMEOUT AR occurs.		
M2 Template	**M2 Description**	
SystemTemplate	This parameter defines the maximum number of trying to send a frame when a TIMEOUT AR occurs (depending on whether retry is configured).	
M2 Parameter		
TransportProtocols::FlexRayTpConnectionControl.maxAr		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context		
FrTp	FrTp/FrTpMultipleConfig/FrTpConnectionControl		
BSW Parameter		**BSW Type**	
FrTpMaxAs		EcucIntegerParamDef	
BSW Description			
This parameter defines the maximum number of trying to send a frame when a TIMEOUT AS occurs.			
M2 Template	**M2 Description**		
SystemTemplate	This parameter defines the maximum number of trying to send a frame when a TIMEOUT AS occurs (depending on whether retry is configured)		
M2 Parameter			
TransportProtocols::FlexRayTpConnectionControl.maxAs			
Mapping Rule			**Mapping Type**
1:1 mapping			full

BSW Module	BSW Context		
FrTp	FrTp/FrTpMultipleConfig/FrTpConnectionControl		
BSW Parameter		**BSW Type**	
FrTpMaxBufferSize		EcucIntegerParamDef	
BSW Description			
Limits the maximal buffer size the FrTp can choose in order to limit the amount of Tx buffer that will be requested at the sender side in a segmented transfer.			
M2 Template	**M2 Description**		
SystemTemplate	This parameter is only relevant when having retry activated. It limits the maximal block size the FrTp can choose in order to limit the amount of Tx buffer that will be requested at the sender side in a segmented transfer.		
M2 Parameter			
TransportProtocols::FlexRayTpConnectionControl.maxBufferSize			
Mapping Rule			**Mapping Type**
1:1 mapping			full

BSW Module	BSW Context	
FrTp	FrTp/FrTpMultipleConfig/FrTpConnectionControl	
BSW Parameter		**BSW Type**
FrTpMaxFCWait		EcucIntegerParamDef
BSW Description		
This parameter defines the maximum number of FlowControl N-PDUs with FlowState "WAIT"		
M2 Template	**M2 Description**	
SystemTemplate	This attribute defines the maximum number of FlowControl N-PDUs with FlowState "WAIT"	
M2 Parameter		

TransportProtocols::FlexRayTpConnectionControl.maxFcWait	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	**BSW Context**	
FrTp	FrTp/FrTpMultipleConfig/FrTpConnectionControl	
BSW Parameter		**BSW Type**
FrTpMaxFrIf		EcucIntegerParamDef
BSW Description		
This parameter defines the maximum number of trying to send a frame when the FrIf returns an error.		
M2 Template	**M2 Description**	
SystemTemplate	This parameter defines the maximum number of trying to send a frame when the FrIf returns an error	
M2 Parameter		
TransportProtocols::FlexRayTpConnectionControl.maxFrIf		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
FrTp	FrTp/FrTpMultipleConfig/FrTpConnectionControl	
BSW Parameter		**BSW Type**
FrTpMaxNbrOfNPduPerCycle		EcucIntegerParamDef
BSW Description		
This parameter is part of the ISO 10681-2 protocol's FlowControl parameter "Bandwidth Control (BC)". It limits the number of N-Pdus the sender is allowed to transmit within a FlexRay cycle.		
M2 Template	**M2 Description**	
SystemTemplate	This parameter limits the number of N-Pdus the sender is allowed to transmit within a FlexRay cycle.	
M2 Parameter		
TransportProtocols::FlexRayTpConnectionControl.maxNumberOfNPduPerCycle		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
FrTp	FrTp/FrTpMultipleConfig/FrTpConnectionControl	
BSW Parameter		**BSW Type**
FrTpMaxRn		EcucIntegerParamDef
BSW Description		
This parameter defines the maximum number of retries (if retry is configured).		
M2 Template	**M2 Description**	
SystemTemplate	This parameter defines the maximum number of retries (if retry is configured for the particular channel).	
M2 Parameter		
TransportProtocols::FlexRayTpConnectionControl.maxRetries		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**
FrTp	FrTp/FrTpMultipleConfig/FrTpConnectionControl

— AUTOSAR CONFIDENTIAL —

BSW Parameter		BSW Type
FrTpSCexp		EcucIntegerParamDef
BSW Description		
This parameter is part of the ISO 10681-2 protocol's FlowControl parameter "Bandwidth Control (BC)". It represents the exponent to calculate the minimum number of "Separation Cycles" the sender has to wait for the next transmission of an FrTp N-Pdu.		
M2 Template	**M2 Description**	
SystemTemplate	Exponent to calculate the minimum number of "Separation Cycles" the sender has to wait for the next transmission of an FrTp N-Pdu.	
M2 Parameter		
TransportProtocols::FlexRayTpConnectionControl.separationCycleExponent		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
FrTp	FrTp/FrTpMultipleConfig/FrTpConnectionControl	
BSW Parameter		**BSW Type**
FrTpTimeBr		EcucFloatParamDef
BSW Description		
This parameter defines the time in seconds the FrTp requires to transmit a corresponding FlowControl Frame. According to ISO 10681-2 this parameter is a performance requirement.		
M2 Template	**M2 Description**	
SystemTemplate	Time (in seconds) until transmission of the next FlowControl N-PDU.	
M2 Parameter		
TransportProtocols::FlexRayTpConnectionControl.timeBr		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
FrTp	FrTp/FrTpMultipleConfig/FrTpConnectionControl	
BSW Parameter		**BSW Type**
FrTpTimeBuffer		EcucFloatParamDef
BSW Description		
This parameter defines the time in seconds of waiting for the next try to get a Tx or Rx buffer.		
M2 Template	**M2 Description**	
SystemTemplate	This parameter defines the time of waiting for the next try to get a Tx or Rx buffer. Specified in seconds.	
M2 Parameter		
TransportProtocols::FlexRayTpConnectionControl.timeBuffer		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
FrTp	FrTp/FrTpMultipleConfig/FrTpConnectionControl	
BSW Parameter		**BSW Type**
FrTpTimeFrIf		EcucFloatParamDef
BSW Description		
This parameter defines the time in seconds of waiting for the next try (if retry is activated) to send via FrIf_Transmit.		
M2 Template	**M2 Description**	

SystemTemplate	This parameter defines the time of waiting for the next try to send. Specified in seconds.
M2 Parameter	
TransportProtocols::FlexRayTpConnectionControl.timeFrif	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	**BSW Context**
FrTp	FrTp/FrTpMultipleConfig/FrTpConnectionControl
BSW Parameter	**BSW Type**
FrTpTimeoutAr	EcucFloatParamDef
BSW Description	
This parameter states the timeout in seconds between the PDU transmit request of the Transport Layer to the FlexRay Interface and the corresponding confirmation of the FlexRay Interface on the receiver side (for FC or AF).	
M2 Template	**M2 Description**
SystemTemplate	This parameter states the timeout between the PDU transmit request of the Transport Layer to the FlexRay Interface and the corresponding confirmation of the FlexRay Interface on the receiver side (for FC or AF). Specified in seconds.
M2 Parameter	
TransportProtocols::FlexRayTpConnectionControl.timeoutAr	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	**BSW Context**
FrTp	FrTp/FrTpMultipleConfig/FrTpConnectionControl
BSW Parameter	**BSW Type**
FrTpTimeoutAs	EcucFloatParamDef
BSW Description	
This parameter specifies the timeout in seconds the FrIf shall confirm a transmitted Pdu to the FrTp.	
M2 Template	**M2 Description**
SystemTemplate	This attribute states the timeout between the PDU transmit request for the first PDU of the group used in the current connection of the Transport Layer to the FlexRay Interface and the corresponding confirmation of the FlexRay Interface
M2 Parameter	
TransportProtocols::FlexRayTpConnectionControl.timeoutAs	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	**BSW Context**
FrTp	FrTp/FrTpMultipleConfig/FrTpConnectionControl
BSW Parameter	**BSW Type**
FrTpTimeoutBs	EcucFloatParamDef
BSW Description	
This parameter defines the timeout in seconds for waiting for an FC or AF on the sender side in a 1:1 connection.	
M2 Template	**M2 Description**
SystemTemplate	This parameter defines the timeout in seconds for waiting for an FC or AF on the sender side in a 1:1 connection.
M2 Parameter	
TransportProtocols::FlexRayTpConnectionControl.timeoutBs	

— AUTOSAR CONFIDENTIAL —

Mapping Rule	Mapping Type
1:1 mapping	full

BSW Module	BSW Context	
FrTp	FrTp/FrTpMultipleConfig/FrTpConnectionControl	
BSW Parameter		**BSW Type**
FrTpTimeoutCr		EcucFloatParamDef
BSW Description		
This parameter defines the timeout value in seconds a receiver is waiting for a CF or a LF.		
M2 Template	**M2 Description**	
SystemTemplate	This parameter defines the timeout value in seconds for waiting for a CF or FF-x (in case of retry) after receiving the last CF or after sending an FC or AF on the receiver side. Specified in seconds.	
M2 Parameter		
TransportProtocols::FlexRayTpConnectionControl.timeoutCr		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
FrTp	FrTp/FrTpMultipleConfig	
BSW Parameter		**BSW Type**
FrTpRxPduPool		EcucParamConfContainerDef
BSW Description		
This container contains all Pdus that are assigned to that Pdu Pool.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
FrTp	FrTp/FrTpMultipleConfig/FrTpRxPduPool	
BSW Parameter		**BSW Type**
FrTpRxPdu		EcucParamConfContainerDef
BSW Description		
Container to hold the PDU parameters. ImplementationType: PduInfoType		
M2 Template	**M2 Description**	
SystemTemplate	Reference to a pool of Data NPdus.	
M2 Parameter		
TransportProtocols::FlexRayTpConnection.rxPduPool		
Mapping Rule		**Mapping Type**
Create container for each NPdu that is referenced by the regarded FlexRayTp Connection.		full

BSW Module	BSW Context
FrTp	FrTp/FrTpMultipleConfig/FrTpRxPduPool/FrTpRxPdu

BSW Parameter	**BSW Type**
FrTpRxPduId	EcucIntegerParamDef
BSW Description	
This is a unique identifier for a received message which is forwarded from the FrIf to the FrTp. ImplementationType: PduIdType	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**

BSW Module	**BSW Context**
FrTp	FrTp/FrTpMultipleConfig/FrTpRxPduPool/FrTpRxPdu
BSW Parameter	**BSW Type**
FrTpRxPduRef	EcucReferenceDef
BSW Description	
Reference to a PDU in the global PDU structure.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**
FrTp	FrTp/FrTpMultipleConfig
BSW Parameter	**BSW Type**
FrTpTxPduPool	EcucParamConfContainerDef
BSW Description	
This container contains all Pdus that are assigned to that Pdu Pool.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**
FrTp	FrTp/FrTpMultipleConfig/FrTpTxPduPool
BSW Parameter	**BSW Type**
FrTpTxPdu	EcucParamConfContainerDef
BSW Description	
Container to hold the PDU parameters. ImplementationType: PduInfoType	
M2 Template	**M2 Description**
SystemTemplate	Reference to a pool of Data NPdus.
M2 Parameter	

TransportProtocols::FlexRayTpConnection.txPduPool	
Mapping Rule	**Mapping Type**
Create container for each NPdu that is referenced by the regarded FlexRayTp Connection.	full

BSW Module	**BSW Context**	
FrTp	FrTp/FrTpMultipleConfig/FrTpTxPduPool/FrTpTxPdu	
BSW Parameter		**BSW Type**
FrTpTxConfirmationPduId		EcucIntegerParamDef
BSW Description		
Handle Id to be used by the FrIf to confirm the transmission of the FrTpTxPdu to the FrIf module.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**

BSW Module	**BSW Context**	
FrTp	FrTp/FrTpMultipleConfig/FrTpTxPduPool/FrTpTxPdu	
BSW Parameter		**BSW Type**
FrTpTxPduRef		EcucReferenceDef
BSW Description		
Reference to a PDU in the global PDU structure.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

D.11 Lin Driver Mapping

BSW Module	**BSW Context**	
Lin	Lin	
BSW Parameter		**BSW Type**
LinGeneral		EcucParamConfContainerDef
BSW Description		
This container contains the parameters related to each LIN Driver Unit.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
Lin	Lin/LinGeneral	
BSW Parameter		**BSW Type**

— AUTOSAR CONFIDENTIAL —

LinDevErrorDetect		EcucBooleanParamDef	
BSW Description			
Switches the Development Error Detection and Notification ON or OFF.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	**BSW Context**		
Lin	Lin/LinGeneral		
BSW Parameter		**BSW Type**	
LinIndex		EcucIntegerParamDef	
BSW Description			
Specifies the InstanceId of this module instance. If only one instance is present it shall have the Id 0.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	**BSW Context**		
Lin	Lin/LinGeneral		
BSW Parameter		**BSW Type**	
LinTimeoutDuration		EcucIntegerParamDef	
BSW Description			
Specifies the maximum number of loops for blocking function until a timeout is raised in short term wait loops			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	**BSW Context**		
Lin	Lin/LinGeneral		
BSW Parameter		**BSW Type**	
LinVersionInfoApi		EcucBooleanParamDef	
BSW Description			
Switches the Lin_GetVersionInfo function ON or OFF.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

— AUTOSAR CONFIDENTIAL —

BSW Module	BSW Context	
Lin	Lin	
BSW Parameter		**BSW Type**
LinGlobalConfig		EcucParamConfContainerDef
BSW Description		
This container contains the global configuration parameter of the Lin driver. This container is a MultipleConfigurationContainer, i.e. this container and its sub-containers exit once per configuration set.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
Lin	Lin/LinGlobalConfig	
BSW Parameter		**BSW Type**
LinChannel		EcucParamConfContainerDef
BSW Description		
This container contains the configuration (parameters) of the LIN Controller(s).		
M2 Template	**M2 Description**	
SystemTemplate	A physical channel is the transmission medium that is used to send and receive information between two communicating ECUs. Each CommunicationCluster has at least one physical channel.	
M2 Parameter		
Fibex::FibexCore::CoreTopology::PhysicalChannel		
Mapping Rule		**Mapping Type**
A LinChannel container is constructed per CommunicationConnector belonging to the CommunicationController associated with the owning Lin Module container		full

BSW Module	BSW Context	
Lin	Lin/LinGlobalConfig/LinChannel	
BSW Parameter		**BSW Type**
LinChannelBaudRate		EcucIntegerParamDef
BSW Description		
Specifies the baud rate of the LIN channel		
M2 Template	**M2 Description**	
SystemTemplate	channels speed in bits per second	
M2 Parameter		
Fibex::FibexCore::CoreTopology::CommunicationCluster.speed		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
Lin	Lin/LinGlobalConfig/LinChannel	
BSW Parameter		**BSW Type**
LinChannelEcuMWakeupSource		EcucSymbolicNameReferenceDef
BSW Description		

This parameter contains a reference to the Wakeup Source for this controller as defined in the ECU State Manager.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
Lin	Lin/LinGlobalConfig/LinChannel	
BSW Parameter		**BSW Type**
LinChannelId		EcucIntegerParamDef
BSW Description		
Identifies the LIN channel. Replaces LIN_CHANNEL_INDEX_NAME from the LIN SWS.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
Implicit from each CommunicationConnector on the ECU representing a LIN channel. Increase the LinChannelId for each LIN channel created on the same CommunicationController, for each CommunicationController start indexing at zero.		local

BSW Module	**BSW Context**	
Lin	Lin/LinGlobalConfig/LinChannel	
BSW Parameter		**BSW Type**
LinChannelWakeupSupport		EcucBooleanParamDef
BSW Description		
Specifies if the LIN hardware channel supports wake up functionality		
M2 Template	**M2 Description**	
SystemTemplate	Specifies if the channel supports wake up functionality	
M2 Parameter		
Fibex::FibexCore::CoreTopology::PhysicalChannel.channelWakeupSupport		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
Lin	Lin/LinGlobalConfig/LinChannel	
BSW Parameter		**BSW Type**
LinClockRef		EcucReferenceDef
BSW Description		
Reference to the LIN clock source configuration, which is set in the MCU driver configuration.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
Lin	Lin/LinGlobalConfig	
BSW Parameter		**BSW Type**
LinDemEventParameterRefs		EcucParamConfContainerDef
BSW Description		
Container for the references to DemEventParameter elements which shall be invoked using the API Dem_ReportErrorStatus API in case the corresponding error occurs. The EventId is taken from the referenced DemEventParameter's DemEventId value. The standardized errors are provided in the container and can be extended by vendor specific error references.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
Lin	Lin/LinGlobalConfig/LinDemEventParameterRefs	
BSW Parameter		**BSW Type**
LIN_E_TIMEOUT		EcucSymbolicNameReferenceDef
BSW Description		
Reference to the DemEventParameter which shall be issued when the error "Timeout caused by hardware error" has occured. If the reference is not configured the error shall be reported as DET error.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

D.12 Lin Interface Mapping

BSW Module	BSW Context	
LinIf	LinIf	
BSW Parameter		**BSW Type**
LinIfGeneral		EcucParamConfContainerDef
BSW Description		
M2 Template	**M2 Description**	
System Template	The CommunicationCluster is the main element to describe the topological connection of communicating ECUs.	
M2 Parameter		
CoreTopology::LinCluster		
Mapping Rule		**Mapping Type**
Container must be created if the ECU is connected to a LIN Cluster		full

BSW Module	BSW Context	
LinIf	LinIf/LinIfGeneral	
BSW Parameter		**BSW Type**

LinIfCancelTransmitSupported	EcucBooleanParamDef
BSW Description	
Global Pre-Compile Switch to reliably prevent the generation of the dummy LinIf_CancelTransmit API.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**
LinIf	LinIf/LinIfGeneral
BSW Parameter	**BSW Type**
LinIfDevErrorDetect	EcucBooleanParamDef
BSW Description	
Switches the Development Error Detection and Notification ON or OFF.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**
LinIf	LinIf/LinIfGeneral
BSW Parameter	**BSW Type**
LinIfMultipleDriversSupported	EcucBooleanParamDef
BSW Description	
States if multiple drivers are included in the LIN Interface or not. The reason for this parameter is to reduce the size of LIN Interface if multiple drivers are not used.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**
LinIf	LinIf/LinIfGeneral
BSW Parameter	**BSW Type**
LinIfMultipleTrcvDriverSupported	EcucBooleanParamDef
BSW Description	
States if multiple transceiver drivers are included in the LIN Interface or not. The reason for this parameter is to reduce the size of LIN Interface if multiple transceiver drivers are not used.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**

	local

BSW Module	BSW Context	
LinIf	LinIf/LinIfGeneral	
BSW Parameter		**BSW Type**
LinIfNcOptionalRequestSupported		EcucBooleanParamDef
BSW Description		
States if the node configuration commands Assign NAD and Conditional Change NAD are supported.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
LinIf	LinIf/LinIfGeneral	
BSW Parameter		**BSW Type**
LinIfPublicCddHeaderFile		EcucStringParamDef
BSW Description		
Defines header files for callback functions which shall be included in case of CDDs. Range of characters is 1.. 32.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
LinIf	LinIf/LinIfGeneral	
BSW Parameter		**BSW Type**
LinIfTpSupported		EcucBooleanParamDef
BSW Description		
States if the TP is included in the LIN Interface or not. The reason for this parameter is to reduce the size of LIN Interface if the TP is not used.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
LinIf	LinIf/LinIfGeneral	
BSW Parameter		**BSW Type**
LinIfTrcvDriverSupported		EcucBooleanParamDef
BSW Description		

States if transceiver drivers are included in the LIN Interface or not. The reason for this parameter is to reduce the size of LIN Interface if transceiver drivers are not used.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
LinIf	LinIf/LinIfGeneral	
BSW Parameter		**BSW Type**
LinIfVersionInfoApi		EcucBooleanParamDef
BSW Description		
Switches the LinIf_GetVersionInfo function ON or OFF.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
LinIf	LinIf	
BSW Parameter		**BSW Type**
LinIfGlobalConfig		EcucParamConfContainerDef
BSW Description		
This container contains the global configuration parameter of the LinIf. It is a MultipleConfigurationContainer, i.e. this container and its sub-containers exit once per configuration set.		
M2 Template	**M2 Description**	
System Template	The CommunicationCluster is the main element to describe the topological connection of communicating ECUs.	
M2 Parameter		
CoreTopology::LinCluster		
Mapping Rule		**Mapping Type**
Container must be created if the ECU is connected to a LIN Cluster		full

BSW Module	**BSW Context**	
LinIf	LinIf/LinIfGlobalConfig	
BSW Parameter		**BSW Type**
LinIfChannel		EcucParamConfContainerDef
BSW Description		
M2 Template	**M2 Description**	
System Template	The connection between the referencing ECU and the referenced channel via the referenced controller.	
M2 Parameter		
CoreTopology::CommunicationConnector		
Mapping Rule		**Mapping Type**

— AUTOSAR CONFIDENTIAL —

Container must be created if the CommunicationConnector belonging to the EC U is connected to a LinChannel.	full

BSW Module	BSW Context	
LinIf	LinIf/LinIfGlobalConfig/LinIfChannel	
BSW Parameter		**BSW Type**
LinIfChannelId		EcucIntegerParamDef
BSW Description		
This parameter holds the unique channel index value. The value shall be the same as the ComM-ChannelId of the ComMChannel referenced by LinIfComMNetworkHandleRef. Implementation Type: NetworkHandleType		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
LinIf	LinIf/LinIfGlobalConfig/LinIfChannel	
BSW Parameter		**BSW Type**
LinIfChannelRef		EcucSymbolicNameReferenceDef
BSW Description		
Reference to the used channel in Lin. Replaces LINIF_CHANNEL_INDEX		
M2 Template	**M2 Description**	
System Template	Reference to the channel to which the ECU is connected.	
M2 Parameter		
CommunicationConnector.physicalChannel		
Mapping Rule		**Mapping Type**
emulate reference from System Description		full

BSW Module	BSW Context	
LinIf	LinIf/LinIfGlobalConfig/LinIfChannel	
BSW Parameter		**BSW Type**
LinIfComMNetworkHandleRef		EcucSymbolicNameReferenceDef
BSW Description		
Unique handle to identify one certain LIN network. Reference to one of the network handles configured for the ComM.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**

BSW Module	BSW Context

LinIf	LinIf/LinIfGlobalConfig/LinIfChannel		
BSW Parameter		**BSW Type**	
LinIfFrame		EcucParamConfContainerDef	
BSW Description			
Generic container for all types of LIN frames. The shortName of this container is used as LinIfFrameName.			
M2 Template	**M2 Description**		
SystemTemplate	One frame can be triggered on different channels. If a frame has no frame triggering, it won't be sent at all. A frame triggering has assigned exactly one frame, which it triggers.		
M2 Parameter			
Fibex::Fibex4Lin::LinCommunication::LinFrameTriggering			
Mapping Rule			**Mapping Type**
Create container for each LinFrameTriggering aggregated by the PhysicalChannel representing the regarded LIN channel.			full

BSW Module	**BSW Context**		
LinIf	LinIf/LinIfGlobalConfig/LinIfChannel/LinIfFrame		
BSW Parameter		**BSW Type**	
LinIfChecksumType		EcucEnumerationParamDef	
BSW Description			
Type of checksum that the frame is using.			
M2 Template	**M2 Description**		
SystemTemplate	Type of checksum that the frame is using.		
M2 Parameter			
Fibex::Fibex4Lin::LinCommunication::LinFrameTriggering.checksum			
Mapping Rule			**Mapping Type**
1:1 mapping			full

BSW Module	**BSW Context**		
LinIf	LinIf/LinIfGlobalConfig/LinIfChannel/LinIfFrame		
BSW Parameter		**BSW Type**	
LinIfFixedFrameSdu		EcucParamConfContainerDef	
BSW Description			
In case this is a fixed frame this is the SDU (response). This container represent an eight byte array. The Byte order shall be MSB first.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	**BSW Context**	
LinIf	LinIf/LinIfGlobalConfig/LinIfChannel/LinIfFrame/LinIfFixedFrameSdu	
BSW Parameter		**BSW Type**
LinIfFixedFrameSduByte		EcucParamConfContainerDef
BSW Description		
This container represents a byte within the 8 byte array.		

M2 Template	M2 Description	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context		
LinIf	LinIf/LinIfGlobalConfig/LinIfChannel/LinIfFrame/LinIfFixedFrameSdu/LinIfFixedFrameSduByte		
BSW Parameter		**BSW Type**	
LinIfFixedFrameSduBytePos		EcucIntegerParamDef	
BSW Description			
Index of the Byte in the SDU (response) 8 byte array.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	BSW Context		
LinIf	LinIf/LinIfGlobalConfig/LinIfChannel/LinIfFrame/LinIfFixedFrameSdu/LinIfFixedFrameSduByte		
BSW Parameter		**BSW Type**	
LinIfFixedFrameSduByteVal		EcucIntegerParamDef	
BSW Description			
Byte value in the SDU (response) 8-byte array.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	BSW Context		
LinIf	LinIf/LinIfGlobalConfig/LinIfChannel/LinIfFrame		
BSW Parameter		**BSW Type**	
LinIfFrameType		EcucEnumerationParamDef	
BSW Description			
Type of frame that is described (e.g. sporadic frame). The sporadic slot is not found among the frame types. A sporadic slot is a set of sporadic frames.			
M2 Template	**M2 Description**		
SystemTemplate			
M2 Parameter			
Mapping Rule			**Mapping Type**
see details in EnumerationLiteralDef descritpions			full

— AUTOSAR CONFIDENTIAL —

BSW Module	BSW Context	
LinIf	LinIf/LinIfGlobalConfig/LinIfChannel/LinIfFrame	
BSW Parameter		**BSW Type**
LinIfLength		EcucIntegerParamDef
BSW Description		
Length of the LIN SDU in bytes.		
M2 Template	**M2 Description**	
System Template	The used length (in bytes) of the referencing frame. Should not be confused with a static byte length reserved for each frame by some platforms (e.g. FlexRay).	
M2 Parameter		
FibexCore::CoreCommunication:Frame.frameLength		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
LinIf	LinIf/LinIfGlobalConfig/LinIfChannel/LinIfFrame	
BSW Parameter		**BSW Type**
LinIfPduDirection		EcucChoiceContainerDef
BSW Description		
Direction of the frame		
M2 Template	**M2 Description**	
SystemTemplate	LIN specific attributes to the FrameTriggering	
M2 Parameter		
Fibex::Fibex4Lin::LinCommunication::LinFrameTriggering		
Mapping Rule		**Mapping Type**
Create container for each existing LinFrame.		full

BSW Module	BSW Context	
LinIf	LinIf/LinIfGlobalConfig/LinIfChannel/LinIfFrame	
BSW Parameter		**BSW Type**
LinIfPid		EcucIntegerParamDef
BSW Description		
Protected ID of the LIN frame. There is no reason to calculate the Parity in run-time.		
M2 Template	**M2 Description**	
System Template	To describe a frames identifier on the communication system, usualy with a fixed identifierValue.	
M2 Parameter		
CoreTopology::PhysicalChannel::LinframeTriggering.identifier		
Mapping Rule		**Mapping Type**
parity needs to be calculated and added based on the identifier value specified in FrameTriggering		full

BSW Module	BSW Context	
LinIf	LinIf/LinIfGlobalConfig/LinIfChannel/LinIfFrame	
BSW Parameter		**BSW Type**
LinIfSubstitutionFrames		EcucParamConfContainerDef
BSW Description		
List of unconditional Frames that can be sent in a sporadic Frame slot.		
M2 Template	**M2 Description**	
SystemTemplate		

M2 Parameter	
Fibex::Fibex4Lin::LinCommunication::LinSporadicFrame.substitutedFrame or Fibex::Fibex4Lin::LinCommunication::LinEventTriggeredFrame.linUnconditionalFrame	
Mapping Rule	**Mapping Type**
emulate reference from System Description	full

BSW Module	BSW Context	
LinIf	LinIf/LinIfGlobalConfig/LinIfChannel/LinIfFrame/LinIfSubstitutionFrames	
BSW Parameter		**BSW Type**
LinIfFramePriority		EcucIntegerParamDef
BSW Description		
Priority of an unconditional frame if used as a sporadic frame.		
M2 Template	**M2 Description**	
SystemTemplate	Reference to a group of unconditional frames that share the same frame slot. In case that more than one of the declared frames needs to be transferred, the one first listed shall be chosen.	
M2 Parameter		
Fibex::Fibex4Lin::LinCommunication::LinSporadicFrame.substitutedFrame		
Mapping Rule		**Mapping Type**
In the System Description the priority is described by the Order of the UnconditionalFrames		full

BSW Module	BSW Context	
LinIf	LinIf/LinIfGlobalConfig/LinIfChannel/LinIfFrame/LinIfSubstitutionFrames	
BSW Parameter		**BSW Type**
LinIfSubstitutionFrameRef		EcucReferenceDef
BSW Description		
Reference to an unconditional Frame that can be sent in a sporadic Frame slot.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**

BSW Module	BSW Context	
LinIf	LinIf/LinIfGlobalConfig/LinIfChannel	
BSW Parameter		**BSW Type**
LinIfGotoSleepConfirmationUL		EcucEnumerationParamDef
BSW Description		
This parameter defines the upper layer (UL) module to which the confirmation of the goto-sleep command shall be sent.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**

— AUTOSAR CONFIDENTIAL —

BSW Module	BSW Context	
LinIf	LinIf/LinIfGlobalConfig/LinIfChannel	
BSW Parameter		**BSW Type**
LinIfMaster		EcucParamConfContainerDef
BSW Description		
Each Master can only be connected to one physical channel. This could be compared to the Node parameter in a LDF file.		
M2 Template	**M2 Description**	
System Template	Describing the properties of the refering ecu as a LIN master.	
M2 Parameter		
Fibex4Lin::LinTopology::LinMaster		
Mapping Rule		**Mapping Type**
Create container if the regarded ECU contains a CommunicationController that is defined as a LinMaster. In the System Template the LinMaster is connected to the LinChannel via a CommunicationConnector.		full

BSW Module	BSW Context	
LinIf	LinIf/LinIfGlobalConfig/LinIfChannel/LinIfMaster	
BSW Parameter		**BSW Type**
LinIfClusterTimeBase		EcucFloatParamDef
BSW Description		
Defines a time-base for one LIN cluster in seconds (normally 0.002, 0.005 or 0.010s).		
M2 Template	**M2 Description**	
System Template	Time base is mandatory for the master. It is not used for slaves. LIN 2.0 Spec states: "The time_base value specifies the used time base in the master node to generate the maximum allowed frame transfer time."	
M2 Parameter		
Fibex4Lin::LinTopology::LinMaster.timeBase		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
LinIf	LinIf/LinIfGlobalConfig/LinIfChannel/LinIfMaster	
BSW Parameter		**BSW Type**
LinIfJitter		EcucFloatParamDef
BSW Description		
The jitter specifies the differences between the maximum and minimum delay from time base tick to the header sending start point in seconds.		
M2 Template	**M2 Description**	
System Template	The jitter value specifies the differences between the maximum and minimum delay from time base start point to the frame header sending start point (falling edge of BREAK signal). Unit: seconds	
M2 Parameter		
Fibex4Lin::LinTopology::LinMaster.timeBaseJitter		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
LinIf	LinIf/LinIfGlobalConfig/LinIfChannel	
BSW Parameter		**BSW Type**

— AUTOSAR CONFIDENTIAL —

LinIfScheduleRequestConfirmationUL		EcucEnumerationParamDef
BSW Description		
This parameter defines the upper layer (UL) module to which the confirmation of the successfully performed schedule table change.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**

BSW Module	**BSW Context**	
LinIf	LinIf/LinIfGlobalConfig/LinIfChannel	
BSW Parameter		**BSW Type**
LinIfScheduleTable		EcucParamConfContainerDef
BSW Description		
Describes a schedule table. Each LinIfChannel may have several schedule tables. Each schedule table can only be connected to one channel.		
M2 Template	**M2 Description**	
System Template	The master task (in the master node) transmits frame headers based on a schedule table. The schedule table specifies the identifiers for each header and the interval between the start of a frame and the start of the following frame.	
M2 Parameter		
Fibex::Fibex4Lin::LinCommunication::LinScheduleTable		
Mapping Rule		**Mapping Type**
Create container for each ScheduleTable that is defined for this channel.		full

BSW Module	**BSW Context**	
LinIf	LinIf/LinIfGlobalConfig/LinIfChannel/LinIfScheduleTable	
BSW Parameter		**BSW Type**
LinIfEntry		EcucParamConfContainerDef
BSW Description		
Describes an entry in the schedule table (also known as Frame Slot).		
M2 Template	**M2 Description**	
System Template	Specification of a sending behavior where the transmission order is predefined, e.g. used on LIN buses	
M2 Parameter		
LinFrameTriggering::RelativelyScheduledTiming		
Mapping Rule		**Mapping Type**
Each RelativelyScheduledTiming element in the System Description requires the creation of a LinIfEntry. RelativelyScheduledTiming.scheduleTablle decides to which schedule table the LinIfEntry belongs.		full

BSW Module	**BSW Context**	
LinIf	LinIf/LinIfGlobalConfig/LinIfChannel/LinIfScheduleTable/LinIfEntry	
BSW Parameter		**BSW Type**
LinIfCollisionResolvingRef		EcucReferenceDef
BSW Description		
Reference to the schedule table, which resolves the collision.		
M2 Template	**M2 Description**	

SystemTemplate	Reference to the schedule table, which resolves a collision.
M2 Parameter	
Fibex::Fibex4Lin::LinCommunication::LinEventTriggeredFrame.collisionResolvingSchedule	
Mapping Rule	**Mapping Type**
Emulate the reference from the System Description.	full

BSW Module	**BSW Context**
LinIf	LinIf/LinIfGlobalConfig/LinIfChannel/LinIfScheduleTable/LinIfEntry
BSW Parameter	**BSW Type**
LinIfDelay	EcucFloatParamDef
BSW Description	
Delay to next entry in schedule table in seconds.	
M2 Template	**M2 Description**
System Template	RRelative delay between this tableEntry and the start of the successor in the schedule table in seconds.
M2 Parameter	
Fibex::Fibex4Lin::LinCommunication::ScheduleTableEntry.delay	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	**BSW Context**
LinIf	LinIf/LinIfGlobalConfig/LinIfChannel/LinIfScheduleTable/LinIfEntry
BSW Parameter	**BSW Type**
LinIfEntryIndex	EcucIntegerParamDef
BSW Description	
Position of the Frame Entry in the Schedule Table. The first entry index in the schedule table is 0.	
M2 Template	**M2 Description**
System Template	Relative position in the schedule table. The first entry index in the schedule table is 0.
M2 Parameter	
Fibex::Fibex4Lin::LinCommunication::ScheduleTableEntry.positionInTable	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	**BSW Context**
LinIf	LinIf/LinIfGlobalConfig/LinIfChannel/LinIfScheduleTable/LinIfEntry
BSW Parameter	**BSW Type**
LinIfFrameRef	EcucReferenceDef
BSW Description	
Reference to the frames that belong to this schedule table entry.	
M2 Template	**M2 Description**
System Template	Specifies the LinFrame that will be transmitted in this frame slot.
M2 Parameter	
Fibex::Fibex4Lin::LinCommunication::ApplicationEntry.frameTriggering	
Mapping Rule	**Mapping Type**
Emulate reference from the System Description	full

BSW Module	**BSW Context**
LinIf	LinIf/LinIfGlobalConfig/LinIfChannel/LinIfScheduleTable

— AUTOSAR CONFIDENTIAL —

BSW Parameter	**BSW Type**
LinIfResumePosition	EcucEnumerationParamDef
BSW Description	
Defines, where a schedule table shall be proceeded in case if it has been interrupted by a RUN-ONCE table.	
M2 Template	**M2 Description**
SystemTemplate	Defines, where a schedule table shall be proceeded in case if it has been interrupted by a run-once table or MRF/SRF.
M2 Parameter	
Fibex::Fibex4Lin::LinCommunication::LinScheduleTable.resumePosition	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	**BSW Context**
LinIf	LinIf/LinIfGlobalConfig/LinIfChannel/LinIfScheduleTable
BSW Parameter	**BSW Type**
LinIfRunMode	EcucEnumerationParamDef
BSW Description	
The schedule table can be executed in two different modes.	
M2 Template	**M2 Description**
System Template	The schedule table can be executed in two different modes.
M2 Parameter	
Fibex::Fibex4Lin::LinCommunication::LinScheduleTable.runMode	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	**BSW Context**
LinIf	LinIf/LinIfGlobalConfig/LinIfChannel/LinIfScheduleTable
BSW Parameter	**BSW Type**
LinIfScheduleMode	EcucEnumerationParamDef
BSW Description	
The schedule table can be executed in three different modes.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**

BSW Module	**BSW Context**
LinIf	LinIf/LinIfGlobalConfig/LinIfChannel/LinIfScheduleTable
BSW Parameter	**BSW Type**
LinIfScheduleTableIndex	EcucIntegerParamDef
BSW Description	
This is the unique index used by upper layers to identify a schedule. Note that the NULL_SCHEDULE for each channel has index 0.	
M2 Template	**M2 Description**
M2 Parameter	

Mapping Rule	Mapping Type
	local

BSW Module	BSW Context	
LinIf	LinIf/LinIfGlobalConfig/LinIfChannel/LinIfScheduleTable	
BSW Parameter		**BSW Type**
LinIfScheduleTableName		EcucStringParamDef
BSW Description		
Optional schedule name used to cross-reference with a LDF. This parameter shall always be accompanied by LIN_IF_SCHEDULE_INDEX.		
M2 Template	**M2 Description**	
System Template	Use longName to generate a name for the context element, which enables it to be ** .	
M2 Parameter		
Fibex::Fibex4Lin::LinCommunication::LinScheduleTable.longName		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
LinIf	LinIf/LinIfGlobalConfig/LinIfChannel	
BSW Parameter		**BSW Type**
LinIfSlave		EcucParamConfContainerDef
BSW Description		
The Node attributes of the Slaves are provided with these parameter. The ShortName of this container is used as LinIfNodeName.		
M2 Template	**M2 Description**	
System Template	Describing the properties of the refering ecu as a LIN slave.	
M2 Parameter		
Fibex4Lin::LinTopology::LinSlave		
Mapping Rule		**Mapping Type**
Create container if the regarded ECU contains a CommunicationController that is defined as a LinSlave. In the System Template the LinSlave is connected to the LinChannel via a CommunicationConnector		full

BSW Module	BSW Context	
LinIf	LinIf/LinIfGlobalConfig/LinIfChannel/LinIfSlave	
BSW Parameter		**BSW Type**
LinIfConfiguredNad		EcucIntegerParamDef
BSW Description		
Definition of the initial node address		
M2 Template	**M2 Description**	
System Template	To distinguish LIN slaves that are used twice or more within the same cluster.	
M2 Parameter		
Fibex4Lin::LinTopology::LinSlave.configuredNad		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context
LinIf	LinIf/LinIfGlobalConfig/LinIfChannel/LinIfSlave

BSW Parameter		BSW Type	
LinIfFunctionId		EcucIntegerParamDef	
BSW Description			
LIN function ID			
M2 Template	**M2 Description**		
SystemTemplate	LIN function ID		
M2 Parameter			
Fibex::Fibex4Lin::LinTopology::LinSlave.functionId			
Mapping Rule			**Mapping Type**
1:1 mapping			full

BSW Module	BSW Context		
LinIf	LinIf/LinIfGlobalConfig/LinIfChannel/LinIfSlave		
BSW Parameter		**BSW Type**	
LinIfProtocolVersion		EcucStringParamDef	
BSW Description			
Defines the LIN Protocol version which is used by the slave.			
M2 Template	**M2 Description**		
System Template	Version specifier for a communication protocol.		
M2 Parameter			
Fibex4Lin::LinTopology::LinSlave.protocolVersion			
Mapping Rule			**Mapping Type**
1:1 mapping			full

BSW Module	BSW Context		
LinIf	LinIf/LinIfGlobalConfig/LinIfChannel/LinIfSlave		
BSW Parameter		**BSW Type**	
LinIfSupplierId		EcucIntegerParamDef	
BSW Description			
LIN Supplier ID			
M2 Template	**M2 Description**		
SystemTemplate	LIN Supplier ID		
M2 Parameter			
Fibex::Fibex4Lin::LinTopology::LinSlave.supplierId			
Mapping Rule			**Mapping Type**
1:1 mapping			full

BSW Module	BSW Context		
LinIf	LinIf/LinIfGlobalConfig/LinIfChannel/LinIfSlave		
BSW Parameter		**BSW Type**	
LinIfVariant		EcucIntegerParamDef	
BSW Description			
Specifies the Variant ID			
M2 Template	**M2 Description**		
SystemTemplate	Specifies the Variant ID		
M2 Parameter			
Fibex::Fibex4Lin::LinTopology::LinSlave.variantId			
Mapping Rule			**Mapping Type**
1:1 mapping			full

— AUTOSAR CONFIDENTIAL —

BSW Module	BSW Context	
LinIf	LinIf/LinIfGlobalConfig/LinIfChannel	
BSW Parameter		**BSW Type**
LinIfStartupState		EcucEnumerationParamDef
BSW Description		
Defines the state of each LIN channel after startup		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
LinIf	LinIf/LinIfGlobalConfig/LinIfChannel	
BSW Parameter		**BSW Type**
LinIfTransceiverDrvConfig		EcucParamConfContainerDef
BSW Description		
This container contains the configuration (parameters) of all addressed LIN transceivers by each underlying LIN Transceiver Driver.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
LinIf	LinIf/LinIfGlobalConfig/LinIfChannel/LinIfTransceiverDrvConfig	
BSW Parameter		**BSW Type**
LinIfTrcvIdRef		EcucSymbolicNameReferenceDef
BSW Description		
Logical handle of the underlying LIN transceiver to be served by the LIN Interface.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**

BSW Module	BSW Context	
LinIf	LinIf/LinIfGlobalConfig/LinIfChannel/LinIfTransceiverDrvConfig	
BSW Parameter		**BSW Type**
LinIfTrcvWakeupNotification		EcucBooleanParamDef
BSW Description		
Selects whether wakeup indication notification is supported. True: Enabled False: Disabled		
M2 Template	**M2 Description**	

M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**	
LinIf	LinIf/LinIfGlobalConfig/LinIfChannel	
BSW Parameter		**BSW Type**
LinIfWakeupConfirmationUL		EcucEnumerationParamDef
BSW Description		
This parameter defines the upper layer (UL) module to which the confirmation of the wake-up shall be sent.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**

BSW Module	**BSW Context**	
LinIf	LinIf/LinIfGlobalConfig	
BSW Parameter		**BSW Type**
LinIfTimeBase		EcucFloatParamDef
BSW Description		
The delay between processing two frames is a multiple of the LIN Interface time-base in seconds.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

D.13 LinNm Mapping

BSW Module	**BSW Context**	
LinNm	LinNm	
BSW Parameter		**BSW Type**
LinNmGlobalConfig		EcucParamConfContainerDef
BSW Description		
This container contains the global configuration parameter of the LinNm.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**

BSW Module	**BSW Context**

— AUTOSAR CONFIDENTIAL —

LinNm	LinNm/LinNmGlobalConfig		
BSW Parameter		**BSW Type**	
LinNmBusSynchronizationEnabled		EcucBooleanParamDef	
BSW Description			
Pre-processor switch for enabling bus synchronization support of the LinNm. This feature is required for NM Coordinator nodes only.			
M2 Template	**M2 Description**		
SystemTemplate	Enables bus synchronization support.		
M2 Parameter			
NetworkManagement::NmEcu.nmBusSynchronizationEnabled			
Mapping Rule			**Mapping Type**
1:1 mapping			

BSW Module	**BSW Context**		
LinNm	LinNm/LinNmGlobalConfig		
BSW Parameter		**BSW Type**	
LinNmChannelConfig		EcucParamConfContainerDef	
BSW Description			
This container contains the channel specific configuration parameter of the LinNm.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	**BSW Context**		
LinNm	LinNm/LinNmGlobalConfig/LinNmChannelConfig		
BSW Parameter		**BSW Type**	
LinNmComMNetworkHandleRef		EcucSymbolicNameReferenceDef	
BSW Description			
This reference points to the unique channel defined by the ComMChannel and provides access to the unique channel index value in ComMChannelId.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	**BSW Context**	
LinNm	LinNm/LinNmGlobalConfig	
BSW Parameter		**BSW Type**
LinNmComControlEnabled		EcucBooleanParamDef
BSW Description		
Pre-processor switch for enabling the Communication Control support.		
M2 Template	**M2 Description**	
SystemTemplate	Enables the Communication Control support.	
M2 Parameter		
NetworkManagement::NmEcu.nmComControlEnabled		

Mapping Rule	Mapping Type
1:1 mapping	full

BSW Module	BSW Context	
LinNm	LinNm/LinNmGlobalConfig	
BSW Parameter		**BSW Type**
LinNmComUserDataSupport		EcucBooleanParamDef
BSW Description		
Pre-processor switch for enabling the NM COM user data support		
M2 Template	**M2 Description**	
SystemTemplate	Enable/disable the user data support.	
M2 Parameter		
FibexCore::CoreCommunication::NmPdu.nmDataInformation		
Mapping Rule		**Mapping Type**
If the nmDataInformation attribute is set to true for NmPdus that are transmitted by the regarded Ecu than this parameter must also be set to true.		full

BSW Module	BSW Context	
LinNm	LinNm/LinNmGlobalConfig	
BSW Parameter		**BSW Type**
LinNmCoordinatorSyncSupport		EcucBooleanParamDef
BSW Description		
Enables/disables the coordinator synchronisation support.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
LinNm	LinNm/LinNmGlobalConfig	
BSW Parameter		**BSW Type**
LinNmDevErrorDetect		EcucBooleanParamDef
BSW Description		
Pre-processor switch for enabling development error detection support.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
LinNm	LinNm/LinNmGlobalConfig	
BSW Parameter		**BSW Type**
LinNmNodeDetectionEnabled		EcucBooleanParamDef
BSW Description		
Pre-processor switch for enabling the Node Detection feature.		

M2 Template	**M2 Description**
SystemTemplate	Enables the Request Repeat Message Request support. Only valid if nmNodeIdEnabled is set to true.
M2 Parameter	
NetworkManagement::NmEcu.nmNodeDetectionEnabled	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	**BSW Context**
LinNm	LinNm/LinNmGlobalConfig
BSW Parameter	**BSW Type**
LinNmNodeIdEnabled	EcucBooleanParamDef
BSW Description	
Pre-processor switch for enabling transmission of the source node identifier in NM messages.	
M2 Template	**M2 Description**
SystemTemplate	Enables the source node identifier.
M2 Parameter	
NetworkManagement::NmEcu.nmNodeIdEnabled	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	**BSW Context**
LinNm	LinNm/LinNmGlobalConfig
BSW Parameter	**BSW Type**
LinNmPassiveModeEnabled	EcucBooleanParamDef
BSW Description	
Pre-processor switch for enabling support of the Passive Mode of the LinNm.	
M2 Template	**M2 Description**
SystemTemplate	Enables support of the Passive Mode.
M2 Parameter	
NetworkManagement::NmEcu.nmPassiveModeEnabled	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	**BSW Context**
LinNm	LinNm/LinNmGlobalConfig
BSW Parameter	**BSW Type**
LinNmRemoteSleepIndicationEnabled	EcucBooleanParamDef
BSW Description	
Pre-processor switch for enabling Remote Sleep Indication support. This feature is required for NM Coordinator nodes only.	
M2 Template	**M2 Description**
SystemTemplate	Switch for enabling remote sleep indication support.
M2 Parameter	
NetworkManagement::NmEcu.nmRemoteSleepIndEnabled	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	**BSW Context**

LinNm	LinNm/LinNmGlobalConfig
BSW Parameter	**BSW Type**
LinNmStateChangeIndEnabled	EcucBooleanParamDef
BSW Description	
Pre-processor switch for enabling the Network Management state change notification.	
M2 Template	**M2 Description**
SystemTemplate	Enables the CAN Network Management state change notification.
M2 Parameter	
NetworkManagement::NmEcu.nmStateChangeIndEnabled	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	**BSW Context**
LinNm	LinNm/LinNmGlobalConfig
BSW Parameter	**BSW Type**
LinNmSynchronizationPointEnabled	EcucBooleanParamDef
BSW Description	
Pre-processor switch for enabling the Synchronize NM feature.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**
LinNm	LinNm/LinNmGlobalConfig
BSW Parameter	**BSW Type**
LinNmUserDataEnabled	EcucBooleanParamDef
BSW Description	
Pre-processor switch for enabling User Data support.	
M2 Template	**M2 Description**
SystemTemplate	Switch for enabling user data support.
M2 Parameter	
NetworkManagement::NmEcu.nmUserDataEnabled	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	**BSW Context**
LinNm	LinNm/LinNmGlobalConfig
BSW Parameter	**BSW Type**
LinNmVersionInfoApi	EcucBooleanParamDef
BSW Description	
Pre-processor switch for enabling version info API support.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

D.14 J1939Tp Mapping

BSW Module	BSW Context
J1939Tp	J1939Tp
BSW Parameter	**BSW Type**
J1939TpConfiguration	EcucParamConfContainerDef
BSW Description	
This container contains the configuration parameters and sub containers of the J1939Tp module that define the communication paths. This container is a MultipleConfigurationContainer, i.e. this container and its sub-containers exist once per configuration set.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	BSW Context
J1939Tp	J1939Tp/J1939TpConfiguration
BSW Parameter	**BSW Type**
J1939TpDemEventParameterRefs	EcucParamConfContainerDef
BSW Description	
Container for the references to DemEventParameter elements which shall be passed to the API Dem_ReportErrorStatus. The EventId is taken from the referenced DemEventParameter's DemEventId value. The standardized errors are provided in the container and can be extended by vendor specific error references.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	BSW Context
J1939Tp	J1939Tp/J1939TpConfiguration/J1939TpDemEventParameterRefs
BSW Parameter	**BSW Type**
J1939TP_E_COMMUNICATION	EcucSymbolicNameReferenceDef
BSW Description	
Reference to the DemEventParameter which shall be issued after successful or unsuccessful communication.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	BSW Context
J1939Tp	J1939Tp/J1939TpConfiguration

BSW Parameter			BSW Type
J1939TpRxChannel			EcucParamConfContainerDef
BSW Description			
This container describes a reception channel of the J1939Tp module. One channel is used for all N-SDUs that share the same source address (SA) and the same destination address (BAM: DA = 0xFF, CMDT: DA != 0xFF).			
M2 Template	**M2 Description**		
SystemTemplate	A J1939TpChannel represents an internal path for the transmission or reception of a Pdu via J1939Tp and describes the sender and the receiver of this particular connection. The J1939Tp module routes a Pdu (J1939 PGN) through the channel.		
M2 Parameter			
TransportProtocols::J1939TpConnection			
Mapping Rule			**Mapping Type**
Create container for each existing J1939TpConnection that is used to transmit a NSdu.			full

BSW Module	BSW Context		
J1939Tp	J1939Tp/J1939TpConfiguration/J1939TpRxChannel		
BSW Parameter			**BSW Type**
J1939TpRxCmNPdu			EcucParamConfContainerDef
BSW Description			
This N-PDU represents the TP.CM frame of a J1939 transport protocol session. TP.CM is used both by BAM and CMDT to initialize the connection. For CMDT, it is also used to abort the connection.			
M2 Template	**M2 Description**		
SystemTemplate	Reference to the Flow Control NPdus that are used in the CMDT (Connection Mode Data Transfer) for TP.CM in both directions. BAM uses one TP.CM (Transport Protocol Command).		
M2 Parameter			
TransportProtocols::J1939TpConnection.flowControl			
Mapping Rule			**Mapping Type**
Information can be derived from a received directlNPdu that is referenced by the J1939TpConnection.			full

BSW Module	BSW Context		
J1939Tp	J1939Tp/J1939TpConfiguration/J1939TpRxChannel/J1939TpRxCmNPdu		
BSW Parameter			**BSW Type**
J1939TpRxCmNPduId			EcucIntegerParamDef
BSW Description			
The N-PDU identifier used for communication with CanIf.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	BSW Context		
J1939Tp	J1939Tp/J1939TpConfiguration/J1939TpRxChannel/J1939TpRxCmNPdu		
BSW Parameter			**BSW Type**
J1939TpRxCmNPduRef			EcucReferenceDef

BSW Description		
Reference to the Pdu object representing the N-PDU.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
J1939Tp	J1939Tp/J1939TpConfiguration/J1939TpRxChannel	
BSW Parameter		**BSW Type**
J1939TpRxDtNPdu		EcucParamConfContainerDef
BSW Description		
This N-PDU represents the TP.DT frame of a J1939 transport protocol session. TP.DT is used both by BAM and CMDT to transfer the contents of an N-SDU.		
M2 Template	**M2 Description**	
SystemTemplate	There are two transport protocols defined for J1939: BAM (Broadcast Announce Message), which is a broadcast protocol, and CMDT (Connection Mode Data Transfer), which is a point-to-point protocol with flow control.	
M2 Parameter		
TransportProtocols::J1939TpConnection.dataNPdu		
Mapping Rule		**Mapping Type**
Information can be derived from a received NPdu that is referenced by the J1939 TpConnection.		full

BSW Module	BSW Context	
J1939Tp	J1939Tp/J1939TpConfiguration/J1939TpRxChannel/J1939TpRxDtNPdu	
BSW Parameter		**BSW Type**
J1939TpRxDtNPduId		EcucIntegerParamDef
BSW Description		
The N-PDU identifier used for communication with CanIf.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
J1939Tp	J1939Tp/J1939TpConfiguration/J1939TpRxChannel/J1939TpRxDtNPdu	
BSW Parameter		**BSW Type**
J1939TpRxDtNPduRef		EcucReferenceDef
BSW Description		
Reference to the Pdu object representing the N-PDU.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**

— AUTOSAR CONFIDENTIAL —

	local

BSW Module	**BSW Context**
J1939Tp	J1939Tp/J1939TpConfiguration/J1939TpRxChannel
BSW Parameter	**BSW Type**
J1939TpRxPg	EcucParamConfContainerDef
BSW Description	
M2 Template	**M2 Description**
	The J1939TpRxPg represents one IPdu that may be transmitted via the containing J1939TpRxChannel.
M2 Parameter	
Mapping Rule	**Mapping Type**
	full

BSW Module	**BSW Context**
J1939Tp	J1939Tp/J1939TpConfiguration/J1939TpRxChannel/J1939TpRxPg
BSW Parameter	**BSW Type**
J1939TpRxDirectNPdu	EcucParamConfContainerDef
BSW Description	
This N-PDU represents the short frame that is used for a dynamic length PGN when it has a length of less that 8 bytes. Please note: This sub container is only necessary when J1939TpRxPgDynLength is TRUE.	
M2 Template	**M2 Description**
SystemTemplate	In case of variable length IPdus (with system signals of variable length), an additional NPdu (with the PGN in the CAN ID) is used for messages with up to 8 bytes.
M2 Parameter	
TransportProtocols::J1939TpConnection.directNPdu	
Mapping Rule	**Mapping Type**
Information can be derived from a received directlNPdu that is referenced by the J1939TpConnection.	full

BSW Module	**BSW Context**
J1939Tp	J1939Tp/J1939TpConfiguration/J1939TpRxChannel/J1939TpRxPg/J1939TpRxDirectNPdu
BSW Parameter	**BSW Type**
J1939TpRxDirectNPduId	EcucIntegerParamDef
BSW Description	
The N-PDU identifier used for communication with CanIf.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

<table>
<tr><th>BSW Module</th><th colspan="3">BSW Context</th></tr>
<tr><td>J1939Tp</td><td colspan="3">J1939Tp/J1939TpConfiguration/J1939TpRxChannel/J1939TpRxPg/J1939TpRxDirectNPdu</td></tr>
<tr><th colspan="2">BSW Parameter</th><th colspan="2">BSW Type</th></tr>
<tr><td colspan="2">J1939TpRxDirectNPduRef</td><td colspan="2">EcucReferenceDef</td></tr>
<tr><th colspan="4">BSW Description</th></tr>
<tr><td colspan="4">Reference to the Pdu object representing the N-PDU.</td></tr>
<tr><th>M2 Template</th><th colspan="3">M2 Description</th></tr>
<tr><td></td><td colspan="3"></td></tr>
<tr><th colspan="4">M2 Parameter</th></tr>
<tr><td colspan="4"></td></tr>
<tr><th colspan="3">Mapping Rule</th><th>Mapping Type</th></tr>
<tr><td colspan="3"></td><td>local</td></tr>
</table>

<table>
<tr><th>BSW Module</th><th colspan="3">BSW Context</th></tr>
<tr><td>J1939Tp</td><td colspan="3">J1939Tp/J1939TpConfiguration/J1939TpRxChannel/J1939TpRxPg</td></tr>
<tr><th colspan="2">BSW Parameter</th><th colspan="2">BSW Type</th></tr>
<tr><td colspan="2">J1939TpRxNSdu</td><td colspan="2">EcucParamConfContainerDef</td></tr>
<tr><th colspan="4">BSW Description</th></tr>
<tr><td colspan="4">This container describes the parameters that are relevant for the reception of a specific N-SDU.</td></tr>
<tr><th>M2 Template</th><th colspan="3">M2 Description</th></tr>
<tr><td>SystemTemplate</td><td colspan="3">Reference to IPdus that are segmented by the Transport Protocol.</td></tr>
<tr><th colspan="4">M2 Parameter</th></tr>
<tr><td colspan="4">TransportProtocols::J1939TpConnection.tpSdu</td></tr>
<tr><th colspan="3">Mapping Rule</th><th>Mapping Type</th></tr>
<tr><td colspan="3">1:1 mapping</td><td>full</td></tr>
</table>

<table>
<tr><th>BSW Module</th><th colspan="3">BSW Context</th></tr>
<tr><td>J1939Tp</td><td colspan="3">J1939Tp/J1939TpConfiguration/J1939TpRxChannel/J1939TpRxPg/J1939TpRxNSdu</td></tr>
<tr><th colspan="2">BSW Parameter</th><th colspan="2">BSW Type</th></tr>
<tr><td colspan="2">J1939TpRxNSduRef</td><td colspan="2">EcucReferenceDef</td></tr>
<tr><th colspan="4">BSW Description</th></tr>
<tr><td colspan="4">Reference to the Pdu object representing the N-SDU.</td></tr>
<tr><th>M2 Template</th><th colspan="3">M2 Description</th></tr>
<tr><td></td><td colspan="3"></td></tr>
<tr><th colspan="4">M2 Parameter</th></tr>
<tr><td colspan="4"></td></tr>
<tr><th colspan="3">Mapping Rule</th><th>Mapping Type</th></tr>
<tr><td colspan="3"></td><td>local</td></tr>
</table>

<table>
<tr><th>BSW Module</th><th colspan="3">BSW Context</th></tr>
<tr><td>J1939Tp</td><td colspan="3">J1939Tp/J1939TpConfiguration/J1939TpRxChannel/J1939TpRxPg</td></tr>
<tr><th colspan="2">BSW Parameter</th><th colspan="2">BSW Type</th></tr>
<tr><td colspan="2">J1939TpRxPgDynLength</td><td colspan="2">EcucBooleanParamDef</td></tr>
<tr><th colspan="4">BSW Description</th></tr>
<tr><td colspan="4">This flag is set to TRUE when the N-SDU refers to a PGN with variable length.

Please note: When this attribute is TRUE, the sub container J1939TpRxDirectNPdu is required.</td></tr>
</table>

<table>
<tr><th>M2 Template</th><th colspan="2">M2 Description</th></tr>
<tr><td>SystemTemplate</td><td colspan="2">The length of dynamic length signals is variable in run-time. Only a maximum length of such a signal is specified in the configuration (attribute length).</td></tr>
<tr><th colspan="3">M2 Parameter</th></tr>
<tr><td colspan="3">Fibex::FibexCore::CoreCommunication::SystemSignal.dynamicLength</td></tr>
<tr><th colspan="2">Mapping Rule</th><th>Mapping Type</th></tr>
<tr><td colspan="2">If a Pdu that is referenced by the J1939TpConnection contains a dynamicLength Signal than set this parameter to true.</td><td>full</td></tr>
</table>

<table>
<tr><th>BSW Module</th><th colspan="2">BSW Context</th></tr>
<tr><td>J1939Tp</td><td colspan="2">J1939Tp/J1939TpConfiguration/J1939TpRxChannel/J1939TpRxPg</td></tr>
<tr><th colspan="2">BSW Parameter</th><th>BSW Type</th></tr>
<tr><td colspan="2">J1939TpRxPgPGN</td><td>EcucIntegerParamDef</td></tr>
<tr><th colspan="3">BSW Description</th></tr>
<tr><td colspan="3">Defines the PGN which is represented by the N-SDU.</td></tr>
<tr><th>M2 Template</th><th colspan="2">M2 Description</th></tr>
<tr><td></td><td colspan="2"></td></tr>
<tr><th colspan="3">M2 Parameter</th></tr>
<tr><td colspan="3"></td></tr>
<tr><th colspan="2">Mapping Rule</th><th>Mapping Type</th></tr>
<tr><td colspan="2"></td><td>local</td></tr>
</table>

<table>
<tr><th>BSW Module</th><th colspan="2">BSW Context</th></tr>
<tr><td>J1939Tp</td><td colspan="2">J1939Tp/J1939TpConfiguration/J1939TpRxChannel</td></tr>
<tr><th colspan="2">BSW Parameter</th><th>BSW Type</th></tr>
<tr><td colspan="2">J1939TpRxProtocolType</td><td>EcucEnumerationParamDef</td></tr>
<tr><th colspan="3">BSW Description</th></tr>
<tr><td colspan="3">Protocol type of the referencing connection, which is either BAM or CMDT.

Please note: The protocol type is determined at configuration time by the PDU format of the PGN of an N-SDU: If the first byte of the PGN is smaller than 0xF0, CMDT will be used, otherwise BAM.</td></tr>
<tr><th>M2 Template</th><th colspan="2">M2 Description</th></tr>
<tr><td>SystemTemplate</td><td colspan="2">BAM (Broadcast Announce Message) is a broadcast protocol. If this attribute is set to true broadcast is used. Since address FF is the only broadcast address, there's no reason to configure it.</td></tr>
<tr><th colspan="3">M2 Parameter</th></tr>
<tr><td colspan="3">TransportProtocols::J1939TpConnection.broadcast</td></tr>
<tr><th colspan="2">Mapping Rule</th><th>Mapping Type</th></tr>
<tr><td colspan="2">If the braodcast attribute is set to true than set this parameter to J1939TP_PROTOCOL_BAM</td><td>full</td></tr>
</table>

<table>
<tr><th>BSW Module</th><th colspan="2">BSW Context</th></tr>
<tr><td>J1939Tp</td><td colspan="2">J1939Tp/J1939TpConfiguration/J1939TpRxChannel</td></tr>
<tr><th colspan="2">BSW Parameter</th><th>BSW Type</th></tr>
<tr><td colspan="2">J1939TpTxFcNPdu</td><td>EcucParamConfContainerDef</td></tr>
<tr><th colspan="3">BSW Description</th></tr>
</table>

— AUTOSAR CONFIDENTIAL —

<table>
<tr><td colspan="3">This N-PDU represents the TP.CM frame that is used in reverse direction for a J1939 transport protocol session using the CMDT protocol type. TP.CM in reverse direction is used for intermediate and final acknowledgement of received data and to abort the connection.

Please note: This sub container is only required when J1939TpRxProtocolType is J1939TP_PROTOCOL_CMDT.</td></tr>
<tr><td>M2 Template</td><td colspan="2">M2 Description</td></tr>
<tr><td>SystemTemplate</td><td colspan="2">Reference to the Flow Control NPdus that are used in the CMDT (Connection Mode Data Transfer) for TP.CM in both directions. BAM uses one TP.CM (Transport Protocol Command).</td></tr>
<tr><td colspan="3">M2 Parameter</td></tr>
<tr><td colspan="3">TransportProtocols::J1939TpConnection.flowControl</td></tr>
<tr><td colspan="2">Mapping Rule</td><td>Mapping Type</td></tr>
<tr><td colspan="2">Information can be derived from a received FlowControlNPdu that is referenced by the J1939TpConnection.</td><td>full</td></tr>
</table>

<table>
<tr><td>BSW Module</td><td colspan="2">BSW Context</td></tr>
<tr><td>J1939Tp</td><td colspan="2">J1939Tp/J1939TpConfiguration/J1939TpRxChannel/J1939TpTxFcNPdu</td></tr>
<tr><td colspan="2">BSW Parameter</td><td>BSW Type</td></tr>
<tr><td colspan="2">J1939TpTxFcNPduRef</td><td>EcucReferenceDef</td></tr>
<tr><td colspan="3">BSW Description</td></tr>
<tr><td colspan="3">Reference to the Pdu object representing the N-PDU.

Please note: When two channels have identical but exchanged source and destination addresses, the Pdu referenced by this parameter is shared with J1939TpTxCmNPduRef of the corresponding J1939TpTxChannel.</td></tr>
<tr><td>M2 Template</td><td colspan="2">M2 Description</td></tr>
<tr><td></td><td colspan="2"></td></tr>
<tr><td colspan="3">M2 Parameter</td></tr>
<tr><td colspan="3"></td></tr>
<tr><td colspan="2">Mapping Rule</td><td>Mapping Type</td></tr>
<tr><td colspan="2"></td><td>local</td></tr>
</table>

<table>
<tr><td>BSW Module</td><td colspan="2">BSW Context</td></tr>
<tr><td>J1939Tp</td><td colspan="2">J1939Tp/J1939TpConfiguration/J1939TpRxChannel/J1939TpTxFcNPdu</td></tr>
<tr><td colspan="2">BSW Parameter</td><td>BSW Type</td></tr>
<tr><td colspan="2">J1939TpTxFcNPduTxConfId</td><td>EcucIntegerParamDef</td></tr>
<tr><td colspan="3">BSW Description</td></tr>
<tr><td colspan="3">The N-PDU identifier used for Tx confirmation from CanIf.</td></tr>
<tr><td>M2 Template</td><td colspan="2">M2 Description</td></tr>
<tr><td></td><td colspan="2"></td></tr>
<tr><td colspan="3">M2 Parameter</td></tr>
<tr><td colspan="3"></td></tr>
<tr><td colspan="2">Mapping Rule</td><td>Mapping Type</td></tr>
<tr><td colspan="2"></td><td>local</td></tr>
</table>

<table>
<tr><td>BSW Module</td><td colspan="2">BSW Context</td></tr>
<tr><td>J1939Tp</td><td colspan="2">J1939Tp/J1939TpConfiguration</td></tr>
<tr><td colspan="2">BSW Parameter</td><td>BSW Type</td></tr>
<tr><td colspan="2">J1939TpTxChannel</td><td>EcucParamConfContainerDef</td></tr>
<tr><td colspan="3">BSW Description</td></tr>
</table>

This container describes a transmission channel of the J1939Tp module. One channel is used for all N-SDUs that share the same source address (SA) and the same destination address (BAM: DA = 0xFF, CMDT: DA != 0xFF).	
M2 Template	**M2 Description**
SystemTemplate	A J1939TpChannel represents an internal path for the transmission or reception of a Pdu via J1939Tp and describes the sender and the receiver of this particular connection. The J1939Tp module routes a Pdu (J1939 PGN) through the channel.
M2 Parameter	
TransportProtocols::J1939TpConnection	
Mapping Rule	**Mapping Type**
Create container for each existing J1939TpConnection that is used to transmit a NSdu.	full

BSW Module	**BSW Context**
J1939Tp	J1939Tp/J1939TpConfiguration/J1939TpTxChannel
BSW Parameter	**BSW Type**
J1939TpRxFcNPdu	EcucParamConfContainerDef
BSW Description	
This N-PDU represents the TP.CM frame that is used in reverse direction for a J1939 transport protocol session using the CMDT protocol type. TP.CM in reverse direction is used for intermediate and final acknowledgement of received data and to abort the connection. Please note: This sub container is only required when J1939TpRxProtocolType is J1939TP_PROTOCOL_CMDT.	
M2 Template	**M2 Description**
SystemTemplate	Reference to the Flow Control NPdus that are used in the CMDT (Connection Mode Data Transfer) for TP.CM in both directions. BAM uses one TP.CM (Transport Protocol Command).
M2 Parameter	
TransportProtocols::J1939TpConnection.flowControl	
Mapping Rule	**Mapping Type**
Information can be derived from a transmitted FlowControlNPdu that is referenced by the J1939TpConnection.	full

BSW Module	**BSW Context**
J1939Tp	J1939Tp/J1939TpConfiguration/J1939TpTxChannel/J1939TpRxFcNPdu
BSW Parameter	**BSW Type**
J1939TpRxFcNPduId	EcucIntegerParamDef
BSW Description	
The N-PDU identifier used for communication with CanIf.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**
J1939Tp	J1939Tp/J1939TpConfiguration/J1939TpTxChannel/J1939TpRxFcNPdu
BSW Parameter	**BSW Type**

J1939TpRxFcNPduRef		EcucReferenceDef
BSW Description		
Reference to the Pdu object representing the N-PDU. Please note: When two channels have identical but exchanged source and destination addresses, the Pdu referenced by this parameter is shared with J1939TpRxCmNPduRef of the corresponding J1939TpRxChannel.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
J1939Tp	J1939Tp/J1939TpConfiguration/J1939TpTxChannel	
BSW Parameter		**BSW Type**
J1939TpTxCmNPdu		EcucParamConfContainerDef
BSW Description		
This N-PDU represents the TP.CM frame of a J1939 transport protocol session. TP.CM is used both by BAM and CMDT to initialize the connection. For CMDT, it is also used to abort the connection.		
M2 Template	**M2 Description**	
SystemTemplate	Reference to the Flow Control NPdus that are used in the CMDT (Connection Mode Data Transfer) for TP.CM in both directions. BAM uses one TP.CM (Transport Protocol Command).	
M2 Parameter		
TransportProtocols::J1939TpConnection.flowControl		
Mapping Rule		**Mapping Type**
Information can be derived from a transmitted FlowControlNPdu that is referenced by the J1939TpConnection.		full

BSW Module	**BSW Context**	
J1939Tp	J1939Tp/J1939TpConfiguration/J1939TpTxChannel/J1939TpTxCmNPdu	
BSW Parameter		**BSW Type**
J1939TpTxCmNPduRef		EcucReferenceDef
BSW Description		
Reference to the Pdu object representing the N-PDU.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
J1939Tp	J1939Tp/J1939TpConfiguration/J1939TpTxChannel/J1939TpTxCmNPdu	
BSW Parameter		**BSW Type**
J1939TpTxCmNPduTxConfId		EcucIntegerParamDef
BSW Description		
The N-PDU identifier used for Tx confirmation from CanIf.		

M2 Template	M2 Description	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
J1939Tp	J1939Tp/J1939TpConfiguration/J1939TpTxChannel	
BSW Parameter		**BSW Type**
J1939TpTxDtNPdu		EcucParamConfContainerDef
BSW Description		
This N-PDU represents the TP.DT frame of a J1939 transport protocol session. TP.DT is used both by BAM and CMDT to transfer the contents of an N-SDU.		
M2 Template	**M2 Description**	
SystemTemplate	There are two transport protocols defined for J1939: BAM (Broadcast Announce Message), which is a broadcast protocol, and CMDT (Connection Mode Data Transfer), which is a point-to-point protocol with flow control.	
M2 Parameter		
TransportProtocols::J1939TpConnection.dataNPdu		
Mapping Rule		**Mapping Type**
Information can be derived from a transmitted NPdu that is referenced by the J1939TpConnection.		full

BSW Module	BSW Context	
J1939Tp	J1939Tp/J1939TpConfiguration/J1939TpTxChannel/J1939TpTxDtNPdu	
BSW Parameter		**BSW Type**
J1939TpTxDtNPduRef		EcucReferenceDef
BSW Description		
Reference to the Pdu object representing the N-PDU.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
J1939Tp	J1939Tp/J1939TpConfiguration/J1939TpTxChannel/J1939TpTxDtNPdu	
BSW Parameter		**BSW Type**
J1939TpTxDtNPduTxConfId		EcucIntegerParamDef
BSW Description		
The N-PDU identifier used for Tx confirmation from CanIf.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

— AUTOSAR CONFIDENTIAL —

<table>
<tr><th>BSW Module</th><th>BSW Context</th></tr>
<tr><td>J1939Tp</td><td>J1939Tp/J1939TpConfiguration/J1939TpTxChannel</td></tr>
<tr><th>BSW Parameter</th><th>BSW Type</th></tr>
<tr><td>J1939TpTxPg</td><td>EcucParamConfContainerDef</td></tr>
<tr><th colspan="2">BSW Description</th></tr>
<tr><td colspan="2"></td></tr>
<tr><th>M2 Template</th><th>M2 Description</th></tr>
<tr><td>SystemTemplate</td><td>A J1939TpConnection represents an internal path for the transmission or reception of a Pdu via J1939Tp and describes the the sender and the receiver of this particular communication. The J1939Tp module routes a Pdu (J1939 PGN) through the connection.</td></tr>
<tr><th colspan="2">M2 Parameter</th></tr>
<tr><td colspan="2">TransportProtocols::J1939TpConnection</td></tr>
<tr><th>Mapping Rule</th><th>Mapping Type</th></tr>
<tr><td>Create container for each existing J1939TpConnection that is used to transmit a NSdu.</td><td>full</td></tr>
</table>

<table>
<tr><th>BSW Module</th><th>BSW Context</th></tr>
<tr><td>J1939Tp</td><td>J1939Tp/J1939TpConfiguration/J1939TpTxChannel/J1939TpTxPg</td></tr>
<tr><th>BSW Parameter</th><th>BSW Type</th></tr>
<tr><td>J1939TpTxDirectNPdu</td><td>EcucParamConfContainerDef</td></tr>
<tr><th colspan="2">BSW Description</th></tr>
<tr><td colspan="2">This N-PDU represents the short frame that is used for a dynamic length PGN when it has a length of less that 8 bytes.

Please note: This sub container is only necessary when J1939TpTxPgDynLength is TRUE.</td></tr>
<tr><th>M2 Template</th><th>M2 Description</th></tr>
<tr><td>SystemTemplate</td><td>In case of variable length IPdus (with system signals of variable length), an additional NPdu (with the PGN in the CAN ID) is used for messages with up to 8 bytes.</td></tr>
<tr><th colspan="2">M2 Parameter</th></tr>
<tr><td colspan="2">TransportProtocols::J1939TpConnection.directNpdu</td></tr>
<tr><th>Mapping Rule</th><th>Mapping Type</th></tr>
<tr><td>Information can be derived from a transmitted directlNPdu that is referenced by the J1939TpConnection.</td><td>full</td></tr>
</table>

<table>
<tr><th>BSW Module</th><th>BSW Context</th></tr>
<tr><td>J1939Tp</td><td>J1939Tp/J1939TpConfiguration/J1939TpTxChannel/J1939TpTxPg/J1939TpTxDirectNPdu</td></tr>
<tr><th>BSW Parameter</th><th>BSW Type</th></tr>
<tr><td>J1939TpTxDirectNPduRef</td><td>EcucReferenceDef</td></tr>
<tr><th colspan="2">BSW Description</th></tr>
<tr><td colspan="2">Reference to the Pdu object representing the N-PDU.</td></tr>
<tr><th>M2 Template</th><th>M2 Description</th></tr>
<tr><td></td><td></td></tr>
<tr><th colspan="2">M2 Parameter</th></tr>
<tr><td colspan="2"></td></tr>
<tr><th>Mapping Rule</th><th>Mapping Type</th></tr>
<tr><td></td><td>local</td></tr>
</table>

BSW Module	BSW Context	
J1939Tp	J1939Tp/J1939TpConfiguration/J1939TpTxChannel/J1939TpTxPg/J1939TpTxDirectNPdu	
BSW Parameter		**BSW Type**
J1939TpTxDirectNPduTxConfId		EcucIntegerParamDef
BSW Description		
The N-PDU identifier used for Tx confirmation from CanIf.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
J1939Tp	J1939Tp/J1939TpConfiguration/J1939TpTxChannel/J1939TpTxPg	
BSW Parameter		**BSW Type**
J1939TpTxNSdu		EcucParamConfContainerDef
BSW Description		
This container describes the parameters that are relevant for the transmission of a specific N-SDU.		
M2 Template	**M2 Description**	
SystemTemplate	Reference to IPdus that are segmented by the Transport Protocol.	
M2 Parameter		
TransportProtocols::J1939TpConnection.tpSdu		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
J1939Tp	J1939Tp/J1939TpConfiguration/J1939TpTxChannel/J1939TpTxPg/J1939TpTxNSdu	
BSW Parameter		**BSW Type**
J1939TpTxNSduId		EcucIntegerParamDef
BSW Description		
The N-SDU identifier used for communication with PduR.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
J1939Tp	J1939Tp/J1939TpConfiguration/J1939TpTxChannel/J1939TpTxPg/J1939TpTxNSdu	
BSW Parameter		**BSW Type**
J1939TpTxNSduRef		EcucReferenceDef
BSW Description		
Reference to the Pdu object representing the N-SDU.		
M2 Template	**M2 Description**	

M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	BSW Context	
J1939Tp	J1939Tp/J1939TpConfiguration/J1939TpTxChannel/J1939TpTxPg	
BSW Parameter		**BSW Type**
J1939TpTxPgDynLength		EcucBooleanParamDef
BSW Description		
This flag is set to TRUE when the N-SDU refers to a PGN with variable length. Please note: When this attribute is TRUE, the sub container J1939TpTxDirectNPdu is required.		
M2 Template	**M2 Description**	
SystemTemplate	The length of dynamic length signals is variable in run-time. Only a maximum length of such a signal is specified in the configuration (attribute length).	
M2 Parameter		
Fibex::FibexCore::CoreCommunication::SystemSignal.dynamicLength		
Mapping Rule		**Mapping Type**
If a Pdu that is referenced by the J1939TpConnection contains a dynamicLength Signal than set this parameter to true.		full

BSW Module	BSW Context	
J1939Tp	J1939Tp/J1939TpConfiguration/J1939TpTxChannel/J1939TpTxPg	
BSW Parameter		**BSW Type**
J1939TpTxPgPGN		EcucIntegerParamDef
BSW Description		
Defines the PGN which is represented by the N-SDU.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
J1939Tp	J1939Tp/J1939TpConfiguration/J1939TpTxChannel	
BSW Parameter		**BSW Type**
J1939TpTxProtocolType		EcucEnumerationParamDef
BSW Description		
Protocol type of the referencing connection, which is either BAM or CMDT. Please note: The protocol type is determined at configuration time by the PDU format of the PGN of an N-SDU: If the first byte of the PGN is smaller than 0xF0, CMDT will be used, otherwise BAM.		
M2 Template	**M2 Description**	
SystemTemplate	BAM (Broadcast Announce Message) is a broadcast protocol. If this attribute is set to true broadcast is used. Since address FF is the only broadcast address, there's no reason to configure it.	
M2 Parameter		

TransportProtocols::J1939TpConnection.broadcast	
Mapping Rule	**Mapping Type**
If the braodcast attribute is set to true than set this parameter to J1939TP_PROTOCOL_BAM	full

BSW Module	**BSW Context**	
J1939Tp	J1939Tp	
BSW Parameter		**BSW Type**
J1939TpGeneral		EcucParamConfContainerDef
BSW Description		
This container describes the general configuration parameters of the J1939Tp module.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
J1939Tp	J1939Tp/J1939TpGeneral	
BSW Parameter		**BSW Type**
J1939TpDevErrorDetect		EcucBooleanParamDef
BSW Description		
Switches the Development Error Detection and Notification.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
J1939Tp	J1939Tp/J1939TpGeneral	
BSW Parameter		**BSW Type**
J1939TpMainFunctionPeriod		EcucFloatParamDef
BSW Description		
Allow to configure the time for the MainFunction (in seconds). Please note: This configuration value shall be equal to the value in the ScheduleManager module.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
J1939Tp	J1939Tp/J1939TpGeneral	
BSW Parameter		**BSW Type**

— AUTOSAR CONFIDENTIAL —

<table>
<tr><td colspan="2">J1939TpMaxPacketsPerBlock</td><td colspan="2">EcucIntegerParamDef</td></tr>
<tr><td colspan="4">BSW Description</td></tr>
<tr><td colspan="4">Maximum number of N-PDUs the J1939Tp shall send before waiting for an authorization to continue transmission of the following TP.DT frames. This parameter is only relevant the transmission of messages via CMDT. For further details on this parameter value see SAE J1939/21.</td></tr>
<tr><td>M2 Template</td><td colspan="3">M2 Description</td></tr>
<tr><td></td><td colspan="3"></td></tr>
<tr><td colspan="4">M2 Parameter</td></tr>
<tr><td colspan="4"></td></tr>
<tr><td colspan="3">Mapping Rule</td><td>Mapping Type</td></tr>
<tr><td colspan="3"></td><td>local</td></tr>
</table>

<table>
<tr><td>BSW Module</td><td colspan="3">BSW Context</td></tr>
<tr><td>J1939Tp</td><td colspan="3">J1939Tp/J1939TpGeneral</td></tr>
<tr><td colspan="2">BSW Parameter</td><td colspan="2">BSW Type</td></tr>
<tr><td colspan="2">J1939TpPacketsPerBlock</td><td colspan="2">EcucIntegerParamDef</td></tr>
<tr><td colspan="4">BSW Description</td></tr>
<tr><td colspan="4">Number of N-PDUs the J1939Tp module allows the sender to send before waiting for an authorization to continue transmission of the following TP.DT frames. This parameter is only relevant for reception of messages via CMDT. For further details on this parameter value see SAE J1939/21.</td></tr>
<tr><td>M2 Template</td><td colspan="3">M2 Description</td></tr>
<tr><td></td><td colspan="3"></td></tr>
<tr><td colspan="4">M2 Parameter</td></tr>
<tr><td colspan="4"></td></tr>
<tr><td colspan="3">Mapping Rule</td><td>Mapping Type</td></tr>
<tr><td colspan="3"></td><td>local</td></tr>
</table>

<table>
<tr><td>BSW Module</td><td colspan="3">BSW Context</td></tr>
<tr><td>J1939Tp</td><td colspan="3">J1939Tp/J1939TpGeneral</td></tr>
<tr><td colspan="2">BSW Parameter</td><td colspan="2">BSW Type</td></tr>
<tr><td colspan="2">J1939TpTxConfTimeout</td><td colspan="2">EcucFloatParamDef</td></tr>
<tr><td colspan="4">BSW Description</td></tr>
<tr><td colspan="4">Timeout in seconds for the CanIf Tx confirmation. After this time the J1939Tp assumes that an N-PDU could not be transmitted.
Please note: The Tx confirmation timeout should be set to a value that enabled detection of a lost Tx confirmation in time, and that ensures that normal transmission delay caused by lower message priority does not lead to an error.</td></tr>
<tr><td>M2 Template</td><td colspan="3">M2 Description</td></tr>
<tr><td></td><td colspan="3"></td></tr>
<tr><td colspan="4">M2 Parameter</td></tr>
<tr><td colspan="4"></td></tr>
<tr><td colspan="3">Mapping Rule</td><td>Mapping Type</td></tr>
<tr><td colspan="3"></td><td>local</td></tr>
</table>

<table>
<tr><td>BSW Module</td><td colspan="3">BSW Context</td></tr>
<tr><td>J1939Tp</td><td colspan="3">J1939Tp/J1939TpGeneral</td></tr>
<tr><td colspan="2">BSW Parameter</td><td colspan="2">BSW Type</td></tr>
<tr><td colspan="2">J1939TpVersionInfoApi</td><td colspan="2">EcucBooleanParamDef</td></tr>
<tr><td colspan="4">BSW Description</td></tr>
<tr><td colspan="4">The function J1939Tp_GetVersionInfo is configurable (On/Off) by this configuration parameter.</td></tr>
</table>

M2 Template	M2 Description	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

D.15 SoAd Mapping

BSW Module	BSW Context		
SoAd	SoAd		
BSW Parameter		**BSW Type**	
SoAdDemEventParameterRefs		EcucParamConfContainerDef	
BSW Description			
Container for the references to DemEventParameter elements which shall be invoked using the API Dem_ReportErrorStatus API in case the corresponding error occurs. The EventId is taken from the referenced DemEventParameter's DemEventId value. The standardized errors are provided in the container and can be extended by vendor specific error references.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	BSW Context		
SoAd	SoAd/SoAdDemEventParameterRefs		
BSW Parameter		**BSW Type**	
SOAD_E_INTR		EcucSymbolicNameReferenceDef	
BSW Description			
Reference to the DemEventParameter which shall be issued when the error "Interrupted system call" has occured.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	BSW Context	
SoAd	SoAd/SoAdDemEventParameterRefs	
BSW Parameter		**BSW Type**
SOAD_E_IO		EcucSymbolicNameReferenceDef
BSW Description		
Reference to the DemEventParameter which shall be issued when the error "Input/output error" has occured.		
M2 Template	**M2 Description**	
M2 Parameter		

Mapping Rule	Mapping Type
	local

BSW Module	BSW Context	
SoAd	SoAd/SoAdDemEventParameterRefs	
BSW Parameter		**BSW Type**
SOAD_E_UPPERBUFF		EcucSymbolicNameReferenceDef
BSW Description		
Reference to the DemEventParameter which shall be issued when the error "No buffer available in upper layer" has occured.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
SoAd	SoAd	
BSW Parameter		**BSW Type**
SoAdDoIpConfig		EcucParamConfContainerDef
BSW Description		
This container contains all global configuration parameters of the DoIP plug-in.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**

BSW Module	BSW Context	
SoAd	SoAd/SoAdDoIpConfig	
BSW Parameter		**BSW Type**
SoAdDoIpAliveCheckResponseTime		EcucFloatParamDef
BSW Description		
This parameter specifies the maximum time that a DoIP entity shall wait for an Alive Check Response after sending an Alive Check Request.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
SoAd	SoAd/SoAdDoIpConfig	
BSW Parameter		**BSW Type**
SoAdDoIpControlTimeout		EcucFloatParamDef
BSW Description		

This parameter specifies the maximum time that the test equipment waits for a response to a previously sent control command.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
SoAd	SoAd/SoAdDoIpConfig	
BSW Parameter		**BSW Type**
SoAdDoIpEid		EcucParamConfContainerDef
BSW Description		
A unique 6-byte DoIp Entity Identification (EID)		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**

BSW Module	**BSW Context**	
SoAd	SoAd/SoAdDoIpConfig/SoAdDoIpEid	
BSW Parameter		**BSW Type**
SoAdDoIpEidByte		EcucParamConfContainerDef
BSW Description		
One byte of the DoIp Entity Identification (EID).		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**

BSW Module	**BSW Context**	
SoAd	SoAd/SoAdDoIpConfig/SoAdDoIpEid/SoAdDoIpEidByte	
BSW Parameter		**BSW Type**
SoAdDoIpEidByteIndex		EcucIntegerParamDef
BSW Description		
Index of the Eid byte array.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**

— AUTOSAR CONFIDENTIAL —

SoAd	SoAd/SoAdDoIpConfig/SoAdDoIpEid/SoAdDoIpEidByte
BSW Parameter	**BSW Type**
SoAdDoIpEidByteValue	EcucIntegerParamDef
BSW Description	
Byte Value at the SoAdDoIpEidByteIndex position in the Eid byte array.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**
SoAd	SoAd/SoAdDoIpConfig
BSW Parameter	**BSW Type**
SoAdDoIpGenericInactiveTime	EcucFloatParamDef
BSW Description	
This parameter specifies the maximum time of inactivity on a TCP_DATA before it is closed.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**
SoAd	SoAd/SoAdDoIpConfig
BSW Parameter	**BSW Type**
SoAdDoIpHostNameOpt	EcucStringParamDef
BSW Description	
Defines the <manufacturer specific> part of the "host name option". Note: WD ISO 13400 implicitly shows 3 parts to the Host Name Option: 1) It is required to start with "DoIP_" 2) There may be a static OEM specific part 3) There may be a dynamic vehicle specific part, e.g. VIN SoAdDoIpHostNameOpt contains parts 1) and 2) only.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**
SoAd	SoAd/SoAdDoIpConfig
BSW Parameter	**BSW Type**
SoAdDoIpInitialInactiveTime	EcucFloatParamDef
BSW Description	
This parameter specifies the maximum time of inactivity directly after a TCP_DATA socket was established. After the specified time without Routing Activation, the TCP_DATA socket is closed.	

M2 Template	M2 Description	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
SoAd	SoAd/SoAdDoIpConfig	
BSW Parameter		**BSW Type**
SoAdDoIpResponseTimeout		EcucFloatParamDef
BSW Description		
This parameter specifies the maximum time after which a DoIP information request must have been processed and the corresponding response must have been sent by the DoIP entity, otherwise the request or the response must be considered lost.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
SoAd	SoAd/SoAdDoIpConfig	
BSW Parameter		**BSW Type**
SoAdDoIpVidAnnounceInterval		EcucFloatParamDef
BSW Description		
This timing parameter specifies the time between the Vehicle Announcement Messages that are sent by DoIP entities after a valid IP address was configured.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
SoAd	SoAd/SoAdDoIpConfig	
BSW Parameter		**BSW Type**
SoAdDoIpVidAnnounceMaxWait		EcucFloatParamDef
BSW Description		
Describes the maximum time a DoIP entity shall wait before sending an Vehicle Identification Response.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

— AUTOSAR CONFIDENTIAL —

BSW Module	BSW Context	
SoAd	SoAd/SoAdDoIpConfig	
BSW Parameter		**BSW Type**
SoAdDoIpVidAnnounceMinWait		EcucFloatParamDef
BSW Description		
Describes the minimum time a DoIP entity shall wait before sending an Vehicle Identification Response.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
SoAd	SoAd/SoAdDoIpConfig	
BSW Parameter		**BSW Type**
SoAdDoIpVidAnnounceNum		EcucIntegerParamDef
BSW Description		
Specifies the number of Vehicle Announcement messages, which the DoIP entity sends after a valid IP address has been configured.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
SoAd	SoAd	
BSW Parameter		**BSW Type**
SoAdDoIpRoute		EcucParamConfContainerDef
BSW Description		
A SoAd_DoIP_Route allocates a PDU ID to a combination of a DoIP source and a DoIP target address.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**

BSW Module	BSW Context	
SoAd	SoAd/SoAdDoIpRoute	
BSW Parameter		**BSW Type**
SoAdDoIpSocketConnectionRef		EcucReferenceDef
BSW Description		
Reference to the used socket connection.		
M2 Template	**M2 Description**	

M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	BSW Context	
SoAd	SoAd/SoAdDoIpRoute	
BSW Parameter		**BSW Type**
SoAdDoIpSourceAddress		EcucIntegerParamDef
BSW Description		
The logical DoIP address of the source entitiy.		
M2 Template	**M2 Description**	
SystemTemplate	The logical DoIP address of the source entitiy.	
M2 Parameter		
Fibex4Ethernet::EthernetCommunication::SocketConnection.doIpSourceAddress		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
SoAd	SoAd/SoAdDoIpRoute	
BSW Parameter		**BSW Type**
SoAdDoIpTargetAddress		EcucIntegerParamDef
BSW Description		
The logical DoIP address of the target entity.		
M2 Template	**M2 Description**	
SystemTemplate	The logical DoIP address of the target entity.	
M2 Parameter		
Fibex4Ethernet::EthernetCommunication::SocketConnection.doIpTargetAddress		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
SoAd	SoAd	
BSW Parameter		**BSW Type**
SoAdGeneral		EcucParamConfContainerDef
BSW Description		
This container contains all global configuration parameters of SoAd configured from the Pdu Router Module perspective.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**

BSW Module	BSW Context	
SoAd	SoAd/SoAdGeneral	
BSW Parameter		**BSW Type**
SoAdBufferMemorySize		EcucIntegerParamDef

BSW Description	
Memory size reserved for SoAd buffers.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	BSW Context	
SoAd	SoAd/SoAdGeneral	
BSW Parameter		**BSW Type**
SoAdCallbackApi		EcucBooleanParamDef
BSW Description		
True if the TCP/IP stack supports the AUTOSAR Call-back API in addition to the Berkeley Socket API. TRUE: TCP/IP Stack supports AUTOSAR callback API FALSE: TCP/IP Stack supports only BSD Sockets.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
SoAd	SoAd/SoAdGeneral	
BSW Parameter		**BSW Type**
SoAdDevErrorDetect		EcucBooleanParamDef
BSW Description		
Pre-processor switch for enabling development error detection support.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
SoAd	SoAd/SoAdGeneral	
BSW Parameter		**BSW Type**
SoAdDoIpActive		EcucBooleanParamDef
BSW Description		
True if a DoIP protocol plug-in is available.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
SoAd	SoAd/SoAdGeneral	
BSW Parameter		**BSW Type**
SoAdDoIpVersionInfoApi		EcucBooleanParamDef
BSW Description		
Switches the DoIP_GetVersionInfo() API ON or OFF.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
SoAd	SoAd/SoAdGeneral	
BSW Parameter		**BSW Type**
SoAdIPv6AddressEnabled		EcucBooleanParamDef
BSW Description		
Allows for increased memory allocation to store IPv6 addresses. TRUE: Enables support for IPv6 addresses FALSE: Only IPv4 addresses are supported		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
SoAd	SoAd/SoAdGeneral	
BSW Parameter		**BSW Type**
SoAdMainFunctionPeriod		EcucFloatParamDef
BSW Description		
Determines the frequency at which the SoAd_MainFunction() is called in [s].		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
SoAd	SoAd/SoAdGeneral	
BSW Parameter		**BSW Type**
SoAdMaxOpenSockets		EcucIntegerParamDef
BSW Description		
Specifies the number of sockets that will be open at any one time.		
M2 Template	**M2 Description**	

M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**	
SoAd	SoAd/SoAdGeneral	
BSW Parameter		**BSW Type**
SoAdPollingInterval		EcucFloatParamDef
BSW Description		
Specifies the interval at which the SoAd shall poll the TCP/IP stack for new information in [s].		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
SoAd	SoAd/SoAdGeneral	
BSW Parameter		**BSW Type**
SoAdSocketCount		EcucIntegerParamDef
BSW Description		
Number of entries in the Socket connection table.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
SoAd	SoAd/SoAdGeneral	
BSW Parameter		**BSW Type**
SoAdTcpIpMainFunctionPeriod		EcucFloatParamDef
BSW Description		
Determines the frequency at which the TcpIp_MainFunctionCyclic() is called in [s].		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
SoAd	SoAd/SoAdGeneral	
BSW Parameter		**BSW Type**
SoAdTcpIpVersionInfoApi		EcucBooleanParamDef
BSW Description		

Activates the TCPIP_GetVersionInfo API. TRUE: Enables the TCPIP_GetVersionInfo API. FALSE: TCPIP_GetVersionInfo API is not included.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
SoAd	SoAd/SoAdGeneral	
BSW Parameter		**BSW Type**
SoAdUdpNmApiEnabled		EcucBooleanParamDef
BSW Description		
Activates the configurable interfaces to be used by UdpNm. TRUE: Enables support for the UdpNm API. FALSE: UdpNm API is not included.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
SoAd	SoAd/SoAdGeneral	
BSW Parameter		**BSW Type**
SoAdVersionInfoApi		EcucBooleanParamDef
BSW Description		
Activates the SoAd_GetVersionInfo() API. TRUE: Enables the SoAd_GetVersionInfo() API. FALSE: SoAd_GetVersionInfo() API is not included.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
SoAd	SoAd/SoAdGeneral	
BSW Parameter		**BSW Type**
SoAdXcpApiEnabled		EcucBooleanParamDef
BSW Description		
Activates the configurable interfaces to be used by Xcp. TRUE: Enables support for the Xcp API. FALSE: Xcp API is not included.		
M2 Template	**M2 Description**	

M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**	
SoAd	SoAd	
BSW Parameter		**BSW Type**
SoAdPduRoute		EcucParamConfContainerDef
BSW Description		
Describes the path of a PDU from the PDU Router to the socket in the TCP/IP stack for transmission.		
M2 Template	**M2 Description**	
SystemTemplate	Ethernet specific attributes to the FrameTriggering	
M2 Parameter		
Fibex::Fibex4Ethernet::EthernetCommunication::EthernetFrameTriggering		
Mapping Rule		**Mapping Type**
Container must be defined for each existing Pdu that is transmitted over the Ethernet by the regarded ECU.		full

BSW Module	**BSW Context**	
SoAd	SoAd/SoAdPduRoute	
BSW Parameter		**BSW Type**
SoAdDestinationId		EcucIntegerParamDef
BSW Description		
ID to be sent on the TCP/IP connection if the PDU header option is enabled.		
M2 Template	**M2 Description**	
SystemTemplate	Identifier is required in case of one port per ECU communication where multiple Frames shall be transmitted over the same connection.	
M2 Parameter		
Fibex::Fibex4Ethernet::SocketConnectionIPduIdentifier::identifier		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
SoAd	SoAd/SoAdPduRoute	
BSW Parameter		**BSW Type**
SoAdDestinationSocketRef		EcucReferenceDef
BSW Description		
Connection on which the PDU is to be sent on, references the appropriate entry in the Socket Connection Table.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**

SoAd	SoAd/SoAdPduRoute		
BSW Parameter		**BSW Type**	
SoAdSourcePduId		EcucIntegerParamDef	
BSW Description			
PDU ID of the PDU coming from the PDU Router.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	**BSW Context**		
SoAd	SoAd/SoAdPduRoute		
BSW Parameter		**BSW Type**	
SoAdSourcePduRef		EcucReferenceDef	
BSW Description			
Reference to the global PDU structure			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	**BSW Context**		
SoAd	SoAd/SoAdPduRoute		
BSW Parameter		**BSW Type**	
SoAdSourceSduLength		EcucIntegerParamDef	
BSW Description			
Length in bytes of the SDU to be sent over the TCP/IP stack.			
M2 Template	**M2 Description**		
SystemTemplate	The used length (in bytes) of the referencing frame. Should not be confused with a static byte length reserved for each frame by some platforms (e.g. FlexRay).		
M2 Parameter			
FibexCore::CoreCommunication::Frame.frameLength			
Mapping Rule			**Mapping Type**
1:1 mapping			full

BSW Module	**BSW Context**	
SoAd	SoAd/SoAdPduRoute	
BSW Parameter		**BSW Type**
SoAdTxConfirmationUL		EcucFunctionNameDef
BSW Description		
This optional parameter defines the name of the <User_TxConfirmation> in case that SoAdUserTxConfirmationUL is configured to Cdd. If SoAdUserTxConfirmationUL equals PduR, Xcp or UdpNm, the name of the <User_TxConfirmation> is fixed and this parameter is skipped. If SoAdUserTxConfirmationUL equals Cdd, the name of the <User_TxConfirmation> is selectable.		
M2 Template	**M2 Description**	

M2 Parameter	
Mapping Rule	**Mapping Type**

BSW Module	**BSW Context**	
SoAd	SoAd/SoAdPduRoute	
BSW Parameter		**BSW Type**
SoAdUserTxConfirmationUL		EcucEnumerationParamDef
BSW Description		
This parameter defines the upper layer (UL) module to which the confirmation of the successfully transmitted SoAdSourcePduId has to be routed via the <User_SoAdTxConfirmation>. This <User_SoAdTxConfirmation> has to be invoked when the confirmation of the configured SoAd-SourcePduId will be received by a Tx confirmation event from the EthIf module.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**

BSW Module	**BSW Context**	
SoAd	SoAd	
BSW Parameter		**BSW Type**
SoAdSocketConnection		EcucParamConfContainerDef
BSW Description		
Information required to receive and transmit data via the TCP/IP stack on a particular connection.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
SoAd	SoAd/SoAdSocketConnection	
BSW Parameter		**BSW Type**
SoAdAutosarConnector		EcucEnumerationParamDef
BSW Description		
Connection point within the AUTOSAR stack for this socket connection Availability of protocol plug-ins. Entries in the Socket and PDU Routing Tables.		
M2 Template	**M2 Description**	
SystemTemplate	Availability of protocol plug-ins. Entries in the Socket and PDU Routing Tables.	
M2 Parameter		
Fibex::Fibex4Ethernet::SocketConnection.autosarConnector		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
SoAd	SoAd/SoAdSocketConnection	
BSW Parameter		**BSW Type**
SoAdDemEventConnectionParameterRefs		EcucParamConfContainerDef
BSW Description		
Container for the references to DemEventParameter elements which shall be invoked using the API Dem_ReportErrorStatus API in case the corresponding error occurs. The EventId is taken from the referenced DemEventParameter's DemEventId value. The standardized errors are provided in the container and can be extended by vendor specific error references.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
SoAd	SoAd/SoAdSocketConnection/SoAdDemEventConnectionParameterRefs	
BSW Parameter		**BSW Type**
SOAD_E_AGAIN		EcucSymbolicNameReferenceDef
BSW Description		
Reference to the DemEventParameter which shall be issued when the error "Resource temporarily unavailable" has occured.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
SoAd	SoAd/SoAdSocketConnection/SoAdDemEventConnectionParameterRefs	
BSW Parameter		**BSW Type**
SOAD_E_CONNABORTED		EcucSymbolicNameReferenceDef
BSW Description		
Reference to the DemEventParameter which shall be issued when the error "Software caused connection abort" has occured.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
SoAd	SoAd/SoAdSocketConnection/SoAdDemEventConnectionParameterRefs	
BSW Parameter		**BSW Type**
SOAD_E_CONNREFUSED		EcucSymbolicNameReferenceDef
BSW Description		
Reference to the DemEventParameter which shall be issued when the error "Connection refused" has occured.		

M2 Template	M2 Description	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context		
SoAd	SoAd/SoAdSocketConnection/SoAdDemEventConnectionParameterRefs		
BSW Parameter		**BSW Type**	
SOAD_E_CONNRESET		EcucSymbolicNameReferenceDef	
BSW Description			
Reference to the DemEventParameter which shall be issued when the error "Connection reset by peer" has occured.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	BSW Context		
SoAd	SoAd/SoAdSocketConnection/SoAdDemEventConnectionParameterRefs		
BSW Parameter		**BSW Type**	
SOAD_E_HOSTDOWN		EcucSymbolicNameReferenceDef	
BSW Description			
Reference to the DemEventParameter which shall be issued when the error "Host is down" has occured.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	BSW Context		
SoAd	SoAd/SoAdSocketConnection/SoAdDemEventConnectionParameterRefs		
BSW Parameter		**BSW Type**	
SOAD_E_HOSTUNREACH		EcucSymbolicNameReferenceDef	
BSW Description			
Reference to the DemEventParameter which shall be issued when the error "Host is down" has occured.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	BSW Context	
SoAd	SoAd/SoAdSocketConnection/SoAdDemEventConnectionParameterRefs	
BSW Parameter		**BSW Type**
SOAD_E_NETDOWN		EcucSymbolicNameReferenceDef
BSW Description		
Reference to the DemEventParameter which shall be issued when the error "Network is down" has occured.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
SoAd	SoAd/SoAdSocketConnection/SoAdDemEventConnectionParameterRefs	
BSW Parameter		**BSW Type**
SOAD_E_NETRESET		EcucSymbolicNameReferenceDef
BSW Description		
Reference to the DemEventParameter which shall be issued when the error "Network dropped connection on reset" has occured.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
SoAd	SoAd/SoAdSocketConnection/SoAdDemEventConnectionParameterRefs	
BSW Parameter		**BSW Type**
SOAD_E_NETUNREACH		EcucSymbolicNameReferenceDef
BSW Description		
Reference to the DemEventParameter which shall be issued when the error "Network is unreachable" has occured.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
SoAd	SoAd/SoAdSocketConnection/SoAdDemEventConnectionParameterRefs	
BSW Parameter		**BSW Type**
SOAD_E_NOTCONN		EcucSymbolicNameReferenceDef
BSW Description		
Reference to the DemEventParameter which shall be issued when the error "Socket is not connected" has occured.		
M2 Template	**M2 Description**	

M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**
SoAd	SoAd/SoAdSocketConnection/SoAdDemEventConnectionParameterRefs
BSW Parameter	**BSW Type**
SOAD_E_PIPE	EcucSymbolicNameReferenceDef
BSW Description	
Reference to the DemEventParameter which shall be issued when the error "Broken pipe" has occured.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**
SoAd	SoAd/SoAdSocketConnection/SoAdDemEventConnectionParameterRefs
BSW Parameter	**BSW Type**
SOAD_E_SDULENGTH	EcucSymbolicNameReferenceDef
BSW Description	
Reference to the DemEventParameter which shall be issued when the error "SDU length mismatch" has occured.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**
SoAd	SoAd/SoAdSocketConnection/SoAdDemEventConnectionParameterRefs
BSW Parameter	**BSW Type**
SOAD_E_TIMEDOUT	EcucSymbolicNameReferenceDef
BSW Description	
Reference to the DemEventParameter which shall be issued when the error "Operation timed out" has occured.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**

SoAd	SoAd/SoAdSocketConnection		
BSW Parameter		**BSW Type**	
SoAdPduHeaderEnable		EcucBooleanParamDef	
BSW Description			
Enables the transmission of the PDU header (ID, length) on this TCP/IP connection. TRUE: Send PDU header before data FALSE: Send data only			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	**BSW Context**		
SoAd	SoAd/SoAdSocketConnection		
BSW Parameter		**BSW Type**	
SoAdPduProvideBufferEnable		EcucBooleanParamDef	
BSW Description			
Enables the use of TP style API towards the PDU Router for this PDU. Will trigger the calls to ProvideRxBuffer and ProvideTxBuffer respectively. TRUE: The TP stype API is to be used towards the PDU Router. FALSE: The IF style API is to be used towards the PDU Router.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	**BSW Context**		
SoAd	SoAd/SoAdSocketConnection		
BSW Parameter		**BSW Type**	
SoAdResourceManagementEnable		EcucBooleanParamDef	
BSW Description			
Enables the resource management option for this socket. May not be activated for UDP sockets in receive and not for DoIP sockets. TRUE: resource management option enabled FALSE: resource management option disabled			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	**BSW Context**

SoAd	SoAd/SoAdSocketConnection		
BSW Parameter		**BSW Type**	
SoAdSocketAutosarApi		EcucBooleanParamDef	
BSW Description			
Enables the use of the AUTOSAR call-back API for this connection. TRUE: Use AUTOSAR call-back API FALSE: Use BSD Socket API Availability of the AUTOSAR Call-back API in the TCP/IP stack.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	**BSW Context**		
SoAd	SoAd/SoAdSocketConnection		
BSW Parameter		**BSW Type**	
SoAdSocketId		EcucIntegerParamDef	
BSW Description			
The Socket ID is used as a reference to a particular connection when transferring data to and from the PDU Router.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	**BSW Context**		
SoAd	SoAd/SoAdSocketConnection		
BSW Parameter		**BSW Type**	
SoAdSocketLocalIpAddress		EcucStringParamDef	
BSW Description			
Local IP address used for this connection. Network configuration. Local and Remote Address need to be in the same subnet.			
M2 Template	**M2 Description**		
SystemTemplate	Logical address that is assigned to devices in a network utilizing the Internet Protocol for communication between its nodes.		
M2 Parameter			
Fibex::Fibex4Ethernet::SocketAddress.ipAddress			
Mapping Rule			**Mapping Type**
SocketConnection contains a localPort reference to the SocketAddress.			full

BSW Module	**BSW Context**	
SoAd	SoAd/SoAdSocketConnection	
BSW Parameter		**BSW Type**
SoAdSocketLocalPort		EcucIntegerParamDef

BSW Description	
Local UDP or TCP port used for this connection.	
M2 Template	**M2 Description**
SystemTemplate	Local Port for TCP/UDP connection. In case the source port is fixed.
M2 Parameter	
Fibex::Fibex4Ethernet::EthernetCommunication::SocketConnection.localPort	
Mapping Rule	**Mapping Type**
This parameter shall be derived from the portAddress attribute that is part of the referenced SocketAddress class	full

BSW Module	BSW Context
SoAd	SoAd/SoAdSocketConnection
BSW Parameter	**BSW Type**
SoAdSocketProtocol	EcucEnumerationParamDef
BSW Description	
Specifies the transport protocol (UDP or TCP).	
M2 Template	**M2 Description**
SystemTemplate	Transport Protocols are responsible for encapsulating application data blocks into datagrams suitable for transfer to the network infrastructure for transmission to the destination host
M2 Parameter	
Fibex::Fibex4Ethernet::SocketConnection.socketProtocol	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	BSW Context
SoAd	SoAd/SoAdSocketConnection
BSW Parameter	**BSW Type**
SoAdSocketRemoteIpAddress	EcucStringParamDef
BSW Description	
IP address where NM packets are being sent to.	
M2 Template	**M2 Description**
SystemTemplate	Logical address that is assigned to devices in a network utilizing the Internet Protocol for communication between its nodes.
M2 Parameter	
Fibex::Fibex4Ethernet::SocketAddress.ipAddress	
Mapping Rule	**Mapping Type**
SocketConnection contains a remotePort reference to the SocketAddress.	full

BSW Module	BSW Context
SoAd	SoAd/SoAdSocketConnection
BSW Parameter	**BSW Type**
SoAdSocketRemotePort	EcucIntegerParamDef
BSW Description	
Remote UDP or TCP port used for this connection.	
M2 Template	**M2 Description**
SystemTemplate	Remote Port for TCP/UDP connection. May be different for each Frame or use the same destination port. In second case identifer attribute needs to be considered.
M2 Parameter	

Fibex::Fibex4Ethernet::EthernetCommunication::SocketConnection.remotePort	
Mapping Rule	**Mapping Type**
This parameter shall be derived from the portAddress attribute that is part of the referenced SocketAddress class	full

BSW Module	**BSW Context**	
SoAd	SoAd/SoAdSocketConnection	
BSW Parameter		**BSW Type**
SoAdSocketTcpInitiate		EcucBooleanParamDef
BSW Description		
Specifies the initiator for this TCP connection. This parameter is only relevant for TCP connections. It will not be defined for UDP sockets. TRUE: This TCP connection is initiated by this module. FALSE: This TCP connection is to be initiated in the listen mode.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
SoAd	SoAd/SoAdSocketConnection	
BSW Parameter		**BSW Type**
SoAdSocketTcpNoDelay		EcucBooleanParamDef
BSW Description		
Specifies not to use the congestion control mechanism for this connection. This parameter is only relevant for TCP connections. It will not be defined for UDP sockets. TRUE: This TCP connection will NOT use congestion control. FALSE: This TCP connection will use congestion control.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
SoAd	SoAd/SoAdSocketConnection	
BSW Parameter		**BSW Type**
SoAdSocketUdpListenOnly		EcucBooleanParamDef
BSW Description		
Used to disable the transmit functionality on this UDP port. This parameter is only relevant for UDP connections. TRUE: This UDP port cannot transmit data FALSE: This UDP port can send and receive data		
M2 Template	**M2 Description**	

M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	BSW Context	
SoAd	SoAd	
BSW Parameter		**BSW Type**
SoAdSocketRoute		EcucParamConfContainerDef
BSW Description		
Describes the path of a PDU from a socket in the TCP/IP stack to the PDU Router after reception in the TCP/IP Stack.		
M2 Template	**M2 Description**	
SystemTemplate	Ethernet specific attributes to the FrameTriggering	
M2 Parameter		
Fibex::Fibex4Ethernet::EthernetCommunication::EthernetFrameTriggering		
Mapping Rule		**Mapping Type**
Container must be defined for each existing Ethernet Pdu that is received by the regarded ECU.		full

BSW Module	BSW Context	
SoAd	SoAd/SoAdSocketRoute	
BSW Parameter		**BSW Type**
SoAdDestinationPduRef		EcucReferenceDef
BSW Description		
Reference to the global PDU structure		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
local		

BSW Module	BSW Context	
SoAd	SoAd/SoAdSocketRoute	
BSW Parameter		**BSW Type**
SoAdDestinationSduLength		EcucIntegerParamDef
BSW Description		
Length in bytes of the data contained in the PDU.		
M2 Template	**M2 Description**	
SystemTemplate	The used length (in bytes) of the referencing frame. Should not be confused with a static byte length reserved for each frame by some platforms (e.g. FlexRay).	
M2 Parameter		
FibexCore::CoreCommunication::Frame.frameLength		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context
SoAd	SoAd/SoAdSocketRoute

<table>
<tr><th colspan="2">BSW Parameter</th><th colspan="2">BSW Type</th></tr>
<tr><td colspan="2">SoAdRxIndicationUL</td><td colspan="2">EcucFunctionNameDef</td></tr>
<tr><th colspan="4">BSW Description</th></tr>
<tr><td colspan="4">This parameter defines the name of the <User_RxIndication> in case that SoAdUserRxIndicationUL is configured to Cdd. If SoAdUserRxIndicationUL equals PduR, Xcp or UdpNm, the name of the <User_RxIndication> is fixed and this parameter is skipped. If SoAdUserRxIndicationUL equals CDD the name of the <User_RxIndication> is selectable.</td></tr>
<tr><th>M2 Template</th><th colspan="3">M2 Description</th></tr>
<tr><td></td><td colspan="3"></td></tr>
<tr><th colspan="4">M2 Parameter</th></tr>
<tr><td colspan="4"></td></tr>
<tr><th colspan="3">Mapping Rule</th><th>Mapping Type</th></tr>
<tr><td colspan="3"></td><td></td></tr>
</table>

<table>
<tr><th>BSW Module</th><th colspan="3">BSW Context</th></tr>
<tr><td>SoAd</td><td colspan="3">SoAd/SoAdSocketRoute</td></tr>
<tr><th colspan="2">BSW Parameter</th><th colspan="2">BSW Type</th></tr>
<tr><td colspan="2">SoAdSourceId</td><td colspan="2">EcucIntegerParamDef</td></tr>
<tr><th colspan="4">BSW Description</th></tr>
<tr><td colspan="4">ID contained in the packet received on the TCP/IP connection if the PDU header option is enabled.</td></tr>
<tr><th>M2 Template</th><th colspan="3">M2 Description</th></tr>
<tr><td>SystemTemplate</td><td colspan="3">Identifier is required in case of one port per ECU communication where multiple Frames shall be transmitted over the same connection.</td></tr>
<tr><th colspan="4">M2 Parameter</th></tr>
<tr><td colspan="4">Fibex::Fibex4Ethernet::SocketConnectionIpduIdentifier.identifier</td></tr>
<tr><th colspan="3">Mapping Rule</th><th>Mapping Type</th></tr>
<tr><td colspan="3">1:1 mapping</td><td>full</td></tr>
</table>

<table>
<tr><th>BSW Module</th><th colspan="3">BSW Context</th></tr>
<tr><td>SoAd</td><td colspan="3">SoAd/SoAdSocketRoute</td></tr>
<tr><th colspan="2">BSW Parameter</th><th colspan="2">BSW Type</th></tr>
<tr><td colspan="2">SoAdSourceSocketRef</td><td colspan="2">EcucReferenceDef</td></tr>
<tr><th colspan="4">BSW Description</th></tr>
<tr><td colspan="4">Connection on which the PDU was received. This references an entry in the Socket Connection Table.</td></tr>
<tr><th>M2 Template</th><th colspan="3">M2 Description</th></tr>
<tr><td></td><td colspan="3"></td></tr>
<tr><th colspan="4">M2 Parameter</th></tr>
<tr><td colspan="4"></td></tr>
<tr><th colspan="3">Mapping Rule</th><th>Mapping Type</th></tr>
<tr><td colspan="3"></td><td>local</td></tr>
</table>

<table>
<tr><th>BSW Module</th><th colspan="3">BSW Context</th></tr>
<tr><td>SoAd</td><td colspan="3">SoAd/SoAdSocketRoute</td></tr>
<tr><th colspan="2">BSW Parameter</th><th colspan="2">BSW Type</th></tr>
<tr><td colspan="2">SoAdUserRxIndicationUL</td><td colspan="2">EcucEnumerationParamDef</td></tr>
<tr><th colspan="4">BSW Description</th></tr>
<tr><td colspan="4">This parameter defines the upper layer (UL) module to which the indication of the successfully received SoAd PDU has to be routed via <User_SoAdRxIndication>. This <User_SoAdRxIndication> has to be invoked when the RX indication is received by the EthIf module.</td></tr>
</table>

M2 Template	M2 Description
M2 Parameter	
Mapping Rule	**Mapping Type**

D.16 UdpNm Mapping

BSW Module	BSW Context
UdpNm	UdpNm
BSW Parameter	**BSW Type**
UdpNmGlobalConfig	EcucParamConfContainerDef
BSW Description	
This container contains all global configuration parameters of UDP NM configured from the NM Module perspective.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**

BSW Module	BSW Context
UdpNm	UdpNm/UdpNmGlobalConfig
BSW Parameter	**BSW Type**
UdpNmBusSynchronizationEnabled	EcucBooleanParamDef
BSW Description	
Pre-processor switch for enabling bus synchronization support. This feature is required for gateway nodes only. It must not be defined if UDPNM_PASSIVE_MODE_ENABLED is defined. This parameter shall be derived from NM_BUS_SYNCHRONIZATION_ENABLED.	
M2 Template	**M2 Description**
SystemTemplate	Enables bus synchronization support.
M2 Parameter	
NetworkManagement::NmEcu.nmBusSynchronizationEnabled	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	BSW Context
UdpNm	UdpNm/UdpNmGlobalConfig
BSW Parameter	**BSW Type**
UdpNmChannelConfig	EcucParamConfContainerDef
BSW Description	
This container contains the channel-specific configuration parameters of the UdpNm.	
M2 Template	**M2 Description**
M2 Parameter	

Mapping Rule	Mapping Type

BSW Module	BSW Context	
UdpNm	UdpNm/UdpNmGlobalConfig/UdpNmChannelConfig	
BSW Parameter		**BSW Type**
UdpNmComMNetworkHandleRef		EcucSymbolicNameReferenceDef
BSW Description		
This reference points to the unique channel defined by the ComMChannel and provides access to the unique channel index value in ComMChannelId.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
UdpNm	UdpNm/UdpNmGlobalConfig/UdpNmChannelConfig	
BSW Parameter		**BSW Type**
UdpNmNodeId		EcucIntegerParamDef
BSW Description		
Node identifier of local node. This parameter is only valid if UDPNM_PASSIVE_MODE_ENABLED is set to OFF and UDPNM_NODE_DETECTION_ENABLED is set to ON.		
M2 Template	**M2 Description**	
SystemTemplate	Node identifier of local NmNode. Must be unique in the NmCluster.	
M2 Parameter		
NetworkManagement::UdpNmNode.nmNodeId		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
UdpNm	UdpNm/UdpNmGlobalConfig/UdpNmChannelConfig	
BSW Parameter		**BSW Type**
UdpNmPduCbvPosition		EcucEnumerationParamDef
BSW Description		

Defines the position of the control bit vector within the NM PACKET.
 	
The value of the parameter represents the location of the control bit vector in the NM PACKET (UDPNM_PDU_BYTE_0 means byte 0, UDPNM_PDU_BYTE_1 means byte 1, UDPNM_PDU_OFF means the control bit vector is not part of the NM PACKET)
 	
See also UDPNM_PDU_NID_POSITION
 	
if (UDPNM_PDU_CBV_POSITION != UDPNM_PDU_OFF && UDPNM_PDU_NID_POSITION != UDPNM_PDU_OFF) then UDPNM_PDU_CBV_POSITION != UDPNM_PDU_NID_POSITION
 	
if (UDPNM_PDU_CBV_POSITION != UDPNM_PDU_OFF && UDPNM_PDU_NID_POSITION == UDPNM_PDU_OFF) then UDPNM_PDU_CBV_POSITION = UDPNM_PDU_BYTE0		

<table>
<tr><th>M2 Template</th><th colspan="2">M2 Description</th></tr>
<tr><td>SystemTemplate</td><td colspan="2">Defines the position of the control bit vector within the NM PDU (Byte positon).</td></tr>
<tr><th colspan="3">M2 Parameter</th></tr>
<tr><td colspan="3">NetworkManagement::UdpNmCluster.nmCbvPosition</td></tr>
<tr><th colspan="2">Mapping Rule</th><th>Mapping Type</th></tr>
<tr><td colspan="2">1:1 mapping</td><td>full</td></tr>
</table>

<table>
<tr><th>BSW Module</th><th colspan="2">BSW Context</th></tr>
<tr><td>UdpNm</td><td colspan="2">UdpNm/UdpNmGlobalConfig/UdpNmChannelConfig</td></tr>
<tr><th colspan="2">BSW Parameter</th><th>BSW Type</th></tr>
<tr><td colspan="2">UdpNmPduLength</td><td>EcucIntegerParamDef</td></tr>
<tr><th colspan="3">BSW Description</th></tr>
<tr><td colspan="3">Defines the length of the NM PACKET in bytes.

Valid values are within the range 0 <= UDPNM_PDU_LENGTH <= 8.</td></tr>
<tr><th>M2 Template</th><th colspan="2">M2 Description</th></tr>
<tr><td>SystemTemplate</td><td colspan="2">Defines the length of the NM PDU (in bytes).</td></tr>
<tr><th colspan="3">M2 Parameter</th></tr>
<tr><td colspan="3">NetworkManagement::NmCluster.nmPduLength</td></tr>
<tr><th colspan="2">Mapping Rule</th><th>Mapping Type</th></tr>
<tr><td colspan="2">1:1 mapping</td><td>full</td></tr>
</table>

<table>
<tr><th>BSW Module</th><th colspan="2">BSW Context</th></tr>
<tr><td>UdpNm</td><td colspan="2">UdpNm/UdpNmGlobalConfig/UdpNmChannelConfig</td></tr>
<tr><th colspan="2">BSW Parameter</th><th>BSW Type</th></tr>
<tr><td colspan="2">UdpNmPduNidPosition</td><td>EcucEnumerationParamDef</td></tr>
<tr><th colspan="3">BSW Description</th></tr>
<tr><td colspan="3">Defines the position of the source node identifier within the NM PACKET.

ImplementationType: UdpNm_PduPositionType

The value of the parameter represents the location of the source node identifier in the NM PACKET (UDPNM_PDU_BYTE_0 means byte 0, UDPNM_PDU_BYTE_1 means byte 1, UDPNM_PDU_OFF means source node identifier is not part of the NM PACKET)

See also UDPNM_PDU_CBV_POSITION

if (UDPNM_PDU_NID_POSITION != UDPNM_PDU_OFF && UDPNM_PDU_CBV_POSITION != UDPNM_PDU_OFF) then UDPNM_PDU_NID_POSITION != UDPNM_PDU_CBV_POSITION

if (UDPNM_PDU_NID_POSITION != UDPNM_PDU_OFF && UDPNM_PDU_CBV_POSITION == UDPNM_PDU_OFF) then UDPNM_PDU_IND_POSITION = UDPNM_PDU_BYTE0</td></tr>
<tr><th>M2 Template</th><th colspan="2">M2 Description</th></tr>
<tr><td>SystemTemplate</td><td colspan="2">Defines the byte position of the source node identifier within the NM PDU.</td></tr>
<tr><th colspan="3">M2 Parameter</th></tr>
<tr><td colspan="3">NetworkManagement::UdpNmCluster.nmNidPosition</td></tr>
<tr><th colspan="2">Mapping Rule</th><th>Mapping Type</th></tr>
<tr><td colspan="2">1:1 mapping</td><td>full</td></tr>
</table>

<table>
<tr><th>BSW Module</th><th>BSW Context</th></tr>
<tr><td>UdpNm</td><td>UdpNm/UdpNmGlobalConfig/UdpNmChannelConfig</td></tr>
</table>

<table>
<tr><th colspan="2">BSW Parameter</th><th colspan="2">BSW Type</th></tr>
<tr><td colspan="2">UdpNmRxPdu</td><td colspan="2">EcucParamConfContainerDef</td></tr>
<tr><th colspan="4">BSW Description</th></tr>
<tr><td colspan="4">This container describes the UdpNm RX PDU's.</td></tr>
<tr><th>M2 Template</th><th colspan="3">M2 Description</th></tr>
<tr><td>SystemTemplate</td><td colspan="3"></td></tr>
<tr><th colspan="4">M2 Parameter</th></tr>
<tr><td colspan="4">NetworkManagement::CanNmNode.rxNmPdu</td></tr>
<tr><th colspan="3">Mapping Rule</th><th>Mapping Type</th></tr>
<tr><td colspan="3">Create container for each NmPdu that is received on the regarded Nm cluster</td><td>full</td></tr>
</table>

<table>
<tr><th>BSW Module</th><th colspan="3">BSW Context</th></tr>
<tr><td>UdpNm</td><td colspan="3">UdpNm/UdpNmGlobalConfig/UdpNmChannelConfig/UdpNmRxPdu</td></tr>
<tr><th colspan="2">BSW Parameter</th><th colspan="2">BSW Type</th></tr>
<tr><td colspan="2">UdpNmRxPduId</td><td colspan="2">EcucIntegerParamDef</td></tr>
<tr><th colspan="4">BSW Description</th></tr>
<tr><td colspan="4">ID of the RxPdu that will be used by a RxIndication of the lower layer.</td></tr>
<tr><th>M2 Template</th><th colspan="3">M2 Description</th></tr>
<tr><td></td><td colspan="3"></td></tr>
<tr><th colspan="4">M2 Parameter</th></tr>
<tr><td colspan="4"></td></tr>
<tr><th colspan="3">Mapping Rule</th><th>Mapping Type</th></tr>
<tr><td colspan="3"></td><td></td></tr>
</table>

<table>
<tr><th>BSW Module</th><th colspan="3">BSW Context</th></tr>
<tr><td>UdpNm</td><td colspan="3">UdpNm/UdpNmGlobalConfig/UdpNmChannelConfig/UdpNmRxPdu</td></tr>
<tr><th colspan="2">BSW Parameter</th><th colspan="2">BSW Type</th></tr>
<tr><td colspan="2">UdpNmRxPduRef</td><td colspan="2">EcucReferenceDef</td></tr>
<tr><th colspan="4">BSW Description</th></tr>
<tr><td colspan="4">The reference to a PDU in the global PDU structure described in the AUTOSAR ECU Configuration Specification. This reference will be used by the UdpNm module to derive the PDU Id.</td></tr>
<tr><th>M2 Template</th><th colspan="3">M2 Description</th></tr>
<tr><td></td><td colspan="3"></td></tr>
<tr><th colspan="4">M2 Parameter</th></tr>
<tr><td colspan="4"></td></tr>
<tr><th colspan="3">Mapping Rule</th><th>Mapping Type</th></tr>
<tr><td colspan="3"></td><td>local</td></tr>
</table>

<table>
<tr><th>BSW Module</th><th colspan="3">BSW Context</th></tr>
<tr><td>UdpNm</td><td colspan="3">UdpNm/UdpNmGlobalConfig/UdpNmChannelConfig</td></tr>
<tr><th colspan="2">BSW Parameter</th><th colspan="2">BSW Type</th></tr>
<tr><td colspan="2">UdpNmTxPdu</td><td colspan="2">EcucParamConfContainerDef</td></tr>
<tr><th colspan="4">BSW Description</th></tr>
<tr><td colspan="4">This container describes the UdpNm TX PDU's.</td></tr>
<tr><th>M2 Template</th><th colspan="3">M2 Description</th></tr>
<tr><td>SystemTemplate</td><td colspan="3"></td></tr>
<tr><th colspan="4">M2 Parameter</th></tr>
<tr><td colspan="4">NetworkManagement::CanNmNode.txNmPdu</td></tr>
<tr><th colspan="3">Mapping Rule</th><th>Mapping Type</th></tr>
<tr><td colspan="3">Create container for each NmPdu that is transmitted on the regarded Nmcluster</td><td>full</td></tr>
</table>

BSW Module	**BSW Context**
UdpNm	UdpNm/UdpNmGlobalConfig/UdpNmChannelConfig/UdpNmTxPdu
BSW Parameter	**BSW Type**
UdpNmTxConfirmationPduId	EcucIntegerParamDef
BSW Description	
Id of the TxPdu that will be used by a TxConfirmation from the lower layer.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**

BSW Module	**BSW Context**
UdpNm	UdpNm/UdpNmGlobalConfig/UdpNmChannelConfig/UdpNmTxPdu
BSW Parameter	**BSW Type**
UdpNmTxPduRef	EcucReferenceDef
BSW Description	
The reference to a PDU in the global PDU structure described in the AUTOSAR ECU Configuration Specification. This reference will be used by the UdpNm module to derive the PDU Id.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**
UdpNm	UdpNm/UdpNmGlobalConfig/UdpNmChannelConfig
BSW Parameter	**BSW Type**
UdpNmUserDataLength	EcucIntegerParamDef
BSW Description	
Defines the length of the user data contained in the NM PACKET. The difference between UDPNM_PDU_LENGTH and applied standardized bytes (source node identifier and control bit vector) within the NM PACKET. Valid values are 0x00..0x08.	
M2 Template	**M2 Description**
SystemTemplate	Defines the length of the user data contained in the NM Pdu.
M2 Parameter	
NetworkManagement::UdpNmCluster.nmUserDataLength	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	**BSW Context**
UdpNm	UdpNm/UdpNmGlobalConfig/UdpNmChannelConfig
BSW Parameter	**BSW Type**
UdpNmUserDataTxPdu	EcucParamConfContainerDef
BSW Description	

This optional container is used to configure the UserNm PDU. This container is only available if UdpNmComUserDataSupport is enabled.		
M2 Template	**M2 Description**	
SystemTemplate	This optional aggregation is used to describe NmUserData that is transmitted in the NmPdu.	
M2 Parameter		
NetworkManagement::CanNmPdu.iSignalToIPduMapping		
Mapping Rule		**Mapping Type**
Create container for each NmPdu that aggregates the ISignalToIPduMapping element. The configuration for these Pdus (e.g. Transfer Properties) shall be derived from this information.		full

BSW Module	**BSW Context**	
UdpNm	UdpNm/UdpNmGlobalConfig/UdpNmChannelConfig/UdpNmUserDataTxPdu	
BSW Parameter		**BSW Type**
UdpNmTxUserDataPduId		EcucIntegerParamDef
BSW Description		
This parameter defines the Handle ID of the NM User Data I-PDU.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
UdpNm	UdpNm/UdpNmGlobalConfig/UdpNmChannelConfig/UdpNmUserDataTxPdu	
BSW Parameter		**BSW Type**
UdpNmTxUserDataPduRef		EcucReferenceDef
BSW Description		
Reference to the NM User Data I-PDU in the global PDU collection.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
UdpNm	UdpNm/UdpNmGlobalConfig/UdpNmChannelConfig	
BSW Parameter		**BSW Type**
UpdNmMainFunctionPeriod		EcucFloatParamDef
BSW Description		
Call cycle of UdpNm_MainFunction_x for the respective instance in [s].		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context
UdpNm	UdpNm/UdpNmGlobalConfig/UdpNmChannelConfig
BSW Parameter	**BSW Type**
UpdNmMsgCycleOffset	EcucFloatParamDef
BSW Description	
Time offset in the periodic transmission node. It determines the start delay of the transmission. $<$ UDPNM_MSG_CYCLE_TIME This parameter is only valid if UDPNM_PASSIVE_MODE_ENABLED is disabled.	
M2 Template	**M2 Description**
SystemTemplate	Node specific time offset in the periodic transmission node. It determines the start delay of the transmission. Specified in seconds.
M2 Parameter	
NetworkManagement::UdpNmNode.nmMsgCycleOffset	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	BSW Context
UdpNm	UdpNm/UdpNmGlobalConfig/UdpNmChannelConfig
BSW Parameter	**BSW Type**
UpdNmMsgCycleTime	EcucFloatParamDef
BSW Description	
Period of a NM-message. It determines the periodic rate in the "periodic transmission mode with bus load reduction" and is the basis for transmit scheduling in the "periodic transmission mode without bus load reduction". NM_TIMEOUT_TIME = n * UDPNM_MSG_CYCLE_TIME This parameter is only valid if UDPNM_PASSIVE_MODE_ENABLED is disabled.	
M2 Template	**M2 Description**
SystemTemplate	Period of a UdpNm message in seconds. It determines the periodic rate in the periodic transmission mode with bus load reduction and is the basis for transmit scheduling in the periodic transmission mode without bus load reduction.
M2 Parameter	
NetworkManagement::UdpNmCluster.nmMsgCycleTime	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	BSW Context
UdpNm	UdpNm/UdpNmGlobalConfig/UdpNmChannelConfig
BSW Parameter	**BSW Type**
UpdNmMsgTimeoutTime	EcucFloatParamDef
BSW Description	
Transmission Timout of NM-message. If there is no transmission confirmation by the UDP Interface within this timeout, the UDPNM module shall gibe an error notification. This parameter is only valid if UDPNM_PASSIVE_MODE_ENABLED is disabled. UDPNM_MSG_TIMEOUT_TIME should be a multiple of UDPNM_MSG_CYCLE_TIME.	
M2 Template	**M2 Description**

SystemTemplate	Timeout of a NM message in seconds. It determines how long the NM shall wait with notification of transmission failure while communication errors occur on the bus.	
M2 Parameter		
NetworkManagement::UdpNmCluster.nmMessageTimeoutTime		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
UdpNm	UdpNm/UdpNmGlobalConfig/UdpNmChannelConfig	
BSW Parameter		**BSW Type**
UpdNmRemoteSleepIndTime		EcucFloatParamDef
BSW Description		
Timeout for Remote Sleep Indication. It defines the time in [s] how long it shall take to recognize that all other nodes are ready to sleep. Typically it should be equal to: n * UDPNM_MSG_CYCLE_TIME, where n denotes the number of NM packets that are normally sent before Remote Sleep Indication is detected. The value of n decremented by one determines the amount of lost NM packets that can be tolerated by the Remote Sleep Indication procedure.		
M2 Template	**M2 Description**	
SystemTemplate	Timeout for Remote Sleep Indication in seconds. It defines the time how long it shall take to recognize that all other nodes are ready to sleep.	
M2 Parameter		
NetworkManagement::UdpNmCluster.nmRepeatMessageTime		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
UdpNm	UdpNm/UdpNmGlobalConfig/UdpNmChannelConfig	
BSW Parameter		**BSW Type**
UpdNmRepeatMessageTime		EcucFloatParamDef
BSW Description		
Timeout for Repeat Message State. It defines the time in [s] how long the NM shall stay in the Repeat Message State. Typically it should be equal to: n * UDPNM_MSG_CYCLE_TIME, where n denotes the number of NM packets that are normally sent in the Repeat Message State. The value of n decremented by one determines the amount of lost NM packets that can be tolerated by the node detection procedure. The value 0 denotes that no Repeat Message State is configured. It means that Repeat Message State is transient what implicates that it is left immediately after entrance and in result no start-up stability is guaranteed and no node detection procedure is possible.		
M2 Template	**M2 Description**	
SystemTemplate	Timeout for Repeat Message State in seconds. Defines the time how long the NM shall stay in the Repeat Message State.	
M2 Parameter		
NetworkManagement::UdpNmCluster.nmRepeatMessageTime		
Mapping Rule		**Mapping Type**
1:1 mapping		full

— AUTOSAR CONFIDENTIAL —

BSW Module	**BSW Context**
UdpNm	UdpNm/UdpNmGlobalConfig/UdpNmChannelConfig
BSW Parameter	**BSW Type**
UpdNmTimeoutTime	EcucFloatParamDef
BSW Description	
Network Timeout for NM packets. It denotes the time in [s] how long the NM shall stay in the Network Mode before transition into Prepare Bus-Sleep Mode shall take place. It shall be equal for all nodes in the cluster. It shall be greater than UDPNM_MSG_CYCLE_TIME. Typically, it should be equal to: x * UDPNM_MSG_CYCLE_TIME, where n denotes the number of NM PACKET cycle times in the Ready Sleep State before transition into the Bus-Sleep Mode is initiated. The value of n decremented by one determines the amount of lost NM packets that can be tolerated by the coordination algorithm.	
M2 Template	**M2 Description**
SystemTemplate	NetworkManagement::UdpNmCluster.nmNetworkTimeout
M2 Parameter	
Network Timeout for UdpNm PDUs in seconds. It denotes the time in [s] how long the NM shall stay in the Network Mode before transition into Prepare Bus-Sleep Mode shall take place.	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	**BSW Context**
UdpNm	UdpNm/UdpNmGlobalConfig/UdpNmChannelConfig
BSW Parameter	**BSW Type**
UpdNmWaitBusSleepTime	EcucFloatParamDef
BSW Description	
Timeout for bus calm down phase. It denotes the time in [s] how long the NM shall stay in the Prepare Bus-Sleep Mode before transition into Bus-Sleep Mode shall take place. It shall be equal for all nodes in the cluster. It shall be long enough to empty all Tx-buffer empty.	
M2 Template	**M2 Description**
SystemTemplate	Timeout for bus calm down phase in seconds. It denotes the time how long the CanNm shall stay in the Prepare Bus-Sleep Mode before transition into Bus-Sleep Mode shall take place.
M2 Parameter	
NetworkManagement::UdpNmCluster.nmWaitBusSleepTime	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	**BSW Context**
UdpNm	UdpNm/UdpNmGlobalConfig
BSW Parameter	**BSW Type**
UdpNmComControl_Enabled	EcucBooleanParamDef
BSW Description	
Pre-processor switch for enabling the Communication Control support. This parameter shall be derived from NM_COM_CONTROL_ENABLED.	

M2 Template	M2 Description	
SystemTemplate	Enables the Communication Control support.	
M2 Parameter		
NetworkManagement::NmEcu.nmComControlEnabled		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context		
UdpNm	UdpNm/UdpNmGlobalConfig		
BSW Parameter		**BSW Type**	
UdpNmComUserDataSupport		EcucBooleanParamDef	
BSW Description			
Enable/disable the user data support.			
M2 Template	**M2 Description**		
SystemTemplate	Enable/disable the user data support.		
M2 Parameter			
FibexCore::CoreCommunication::NmPdu.nmDataInformation			
Mapping Rule			**Mapping Type**
If the nmDataInformation attribute is set to true for NmPdus that are transmitted by the regarded Ecu than this parameter must also be set to true.			full

BSW Module	BSW Context		
UdpNm	UdpNm/UdpNmGlobalConfig		
BSW Parameter		**BSW Type**	
UdpNmCoordinatorEnabled		EcucBooleanParamDef	
BSW Description			
Enable/disable the NM Coordination algorithm to being able to initiate the synchronization algorithm. TRUE: Option is enabled FALSE: The parameter shall be FALSE by default and shall only be allowed to be TRUE if the parameter UDPNM_REMOTE_SLEEP_IND_ENABLED is TRUE.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**

BSW Module	BSW Context	
UdpNm	UdpNm/UdpNmGlobalConfig	
BSW Parameter		**BSW Type**
UdpNmCoordinatorId		EcucIntegerParamDef
BSW Description		
Set the NM coordination ID for this gateway. 0x00: passive coordinator only 0x01 - 0x03: coordinator priority Only valid, if UDPNM_COORDINATOR_ENABLED is TRUE.		
M2 Template	**M2 Description**	

M2 Parameter	
Mapping Rule	**Mapping Type**

BSW Module	**BSW Context**	
UdpNm	UdpNm/UdpNmGlobalConfig	
BSW Parameter		**BSW Type**
UdpNmCoordinatorSyncSupport		EcucBooleanParamDef
BSW Description		
Enables/disables the coordinator synchronisation support.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
UdpNm	UdpNm/UdpNmGlobalConfig	
BSW Parameter		**BSW Type**
UdpNmDemEventParameterRefs		EcucParamConfContainerDef
BSW Description		
Container for the references to DemEventParameter elements which shall be invoked using the API Dem_ReportErrorStatus API in case the corresponding error occurs. The EventId is taken from the referenced DemEventParameter's DemEventId value. The standardized errors are provided in the container and can be extended by vendor specific error references.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
UdpNm	UdpNm/UdpNmGlobalConfig/UdpNmDemEventParameterRefs	
BSW Parameter		**BSW Type**
UDPNM_E_INIT_FAILED		EcucSymbolicNameReferenceDef
BSW Description		
Reference to the DemEventParameter which shall be issued when the error "UdpNm initialization has failed, e.g. selected configuration set doesn't exist" has occured.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
UdpNm	UdpNm/UdpNmGlobalConfig/UdpNmDemEventParameterRefs	
BSW Parameter		**BSW Type**
UDPNM_E_NETWORK_TIMEOUT		EcucSymbolicNameReferenceDef
BSW Description		
Reference to the DemEventParameter which shall be issued when the error "NM-Timeout Timer has abnormally expired outside of the Ready Sleep State" has occured.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
UdpNm	UdpNm/UdpNmGlobalConfig/UdpNmDemEventParameterRefs	
BSW Parameter		**BSW Type**
UDPNM_E_TCPIP_TRANSMIT_ERROR		EcucSymbolicNameReferenceDef
BSW Description		
Reference to the DemEventParameter which shall be issued when the error "A call to the TCP/IP stack has failedA call to the TCP/IP stack has failed" has occured.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
UdpNm	UdpNm/UdpNmGlobalConfig	
BSW Parameter		**BSW Type**
UdpNmDevErrorDetect		EcucBooleanParamDef
BSW Description		
Pre-processor switch for enabling development error detection support.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
UdpNm	UdpNm/UdpNmGlobalConfig	
BSW Parameter		**BSW Type**
UdpNmImmediateRestartEnabled		EcucBooleanParamDef
BSW Description		
Pre-processor switch for enabling the asynchronous transmission of a NM PACKET upon bus-communication request in Prepare-Bus-Sleep mode. Must not be defined if UDPNM_PASSIVE_MODE_ENABLED is defined.		

M2 Template	M2 Description	
SystemTemplate	Enables the asynchronous transmission of a CanNm PDU upon bus-communication request in Prepare-Bus-Sleep mode.	
M2 Parameter		
NetworkManagement::NmClusterCoupling.nmImmediateRestartEnabled		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
UdpNm	UdpNm/UdpNmGlobalConfig	
BSW Parameter		**BSW Type**
UdpNmNodeDetectionEnabled		EcucBooleanParamDef
BSW Description		
Pre-processor switch for enabling the node detection support. This parameter shall be derived from NM_NODE_DETECTION_ENABLED. This parameter shall only be enabled if UDPNM_NODE_ID_ENABLED is defined. If(UdpNmPduCbvPosition != UDPNM_PDU_OFF) then Equal(NmNodeDetectionEnabled) else Equal(False).		
M2 Template	**M2 Description**	
SystemTemplate	Enables the Request Repeat Message Request support. Only valid if nmNodeIdEnabled is set to true.	
M2 Parameter		
NetworkManagement::NmEcu.nmNodeDetectionEnabled		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
UdpNm	UdpNm/UdpNmGlobalConfig	
BSW Parameter		**BSW Type**
UdpNmNodeIdEnabled		EcucBooleanParamDef
BSW Description		
Pre-processor switch for enabling the source node identifier. This parameter shall be derived from NM_NODE_ID_ENABLED.		
M2 Template	**M2 Description**	
SystemTemplate	Enables the source node identifier.	
M2 Parameter		
NetworkManagement::NmEcu.nmNodeIdEnabled		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
UdpNm	UdpNm/UdpNmGlobalConfig	
BSW Parameter		**BSW Type**
UdpNmNumberOfChannels		EcucIntegerParamDef
BSW Description		
Number of NM channels allowed within one ECU.		
M2 Template	**M2 Description**	

— AUTOSAR CONFIDENTIAL —

M2 Parameter	
Mapping Rule	**Mapping Type**

BSW Module	BSW Context	
UdpNm	UdpNm/UdpNmGlobalConfig	
BSW Parameter		**BSW Type**
UdpNmPassiveModeEnabled		EcucBooleanParamDef
BSW Description		
Pre-processor switch for enabling support of the Passive Mode. This parameter shall be derived from NM_PASSIVE_MODE_ENABLED.		
M2 Template	**M2 Description**	
SystemTemplate	Enables support of the Passive Mode.	
M2 Parameter		
NetworkManagement::NmEcu.nmPassiveModeEnabled		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
UdpNm	UdpNm/UdpNmGlobalConfig	
BSW Parameter		**BSW Type**
UdpNmPduRxIndicationEnabled		EcucBooleanParamDef
BSW Description		
Pre-processor switch for enabling the PDU Rx Indication. This parameter shall be derived from NM_PDU_RX_INDICATION_ENABLED.		
M2 Template	**M2 Description**	
SystemTemplate	Switch for enabling the PDU Rx Indication.	
M2 Parameter		
NetworkManagement::NmEcu.nmPduRxIndicationEnabled		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
UdpNm	UdpNm/UdpNmGlobalConfig	
BSW Parameter		**BSW Type**
UdpNmRemoteSleepIndEnabled		EcucBooleanParamDef
BSW Description		
Pre-processor switch for enabling remote sleep indication support. This feature is required for gateway nodes only. It must not be defined if UDPNM_PASSIVE_MODE_ENABLED is defined. This parameter shall be derived from NM_REMOTE_SLEEP_IND_ENABLED.		
M2 Template	**M2 Description**	
SystemTemplate	Switch for enabling remote sleep indication support.	
M2 Parameter		
NetworkManagement::NmEcu.nmRemoteSleepIndEnabled		
Mapping Rule		**Mapping Type**
1:1 mapping		full

— AUTOSAR CONFIDENTIAL —

BSW Module	BSW Context	
UdpNm	UdpNm/UdpNmGlobalConfig	
BSW Parameter		**BSW Type**
UdpNmRepeatMsgIndEnabled		EcucBooleanParamDef
BSW Description		
Enable/disable the notification that a RepeatMessageRequest bit has been received. This parameter shall be derived from NM_REPEAT_MSG_IND_ENABLED.		
M2 Template	**M2 Description**	
SystemTemplate	Switch for enabling the Repeat Message Bit Indication.	
M2 Parameter		
NetworkManagement::NmEcu.nmRepeatMsgIndEnabled		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
UdpNm	UdpNm/UdpNmGlobalConfig	
BSW Parameter		**BSW Type**
UdpNmStateChangeIndEnabled		EcucBooleanParamDef
BSW Description		
Pre-processor switch for enabling the UDP NM state change notification. This parameter shall be derived from NM_STATE_CHANGE_ID_ENABLED.		
M2 Template	**M2 Description**	
SystemTemplate	Enables the CAN Network Management state change notification.	
M2 Parameter		
NetworkManagement::NmEcu.nmStateChangeIndEnabled		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
UdpNm	UdpNm/UdpNmGlobalConfig	
BSW Parameter		**BSW Type**
UdpNmUserDataEnabled		EcucBooleanParamDef
BSW Description		
Pre-processor switch for enabling user data support. This parameter shall be derived from NM_USER_DATA_ENABLED.		
M2 Template	**M2 Description**	
SystemTemplate	Switch for enabling user data support.	
M2 Parameter		
NetworkManagement::NmEcu.nmUserDataEnabled		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
UdpNm	UdpNm/UdpNmGlobalConfig	
BSW Parameter		**BSW Type**
UdpNmVersionInfoApi		EcucBooleanParamDef
BSW Description		

Pre-processor switch for enabling version info API support.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

D.17 Com Mapping

BSW Module	**BSW Context**	
Com	Com	
BSW Parameter		**BSW Type**
ComConfig		EcucParamConfContainerDef
BSW Description		
This container contains the configuration parameters and sub containers of the AUTOSAR COM module. This container is a MultipleConfigurationContainer, i.e. this container and its sub-containers exist once per configuration set.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
Com	Com/ComConfig	
BSW Parameter		**BSW Type**
ComConfigurationId		EcucIntegerParamDef
BSW Description		
This ID is returned by a call to Com_GetConfigurationId.		
M2 Template	**M2 Description**	
SystemTemplate	This ID is returned by a call to Com GetConfigurationId()	
M2 Parameter		
Fibex::FibexCore::CoreTopology::ECUInstance::ComConfigurationID		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
Com	Com/ComConfig	
BSW Parameter		**BSW Type**
ComGwMapping		EcucParamConfContainerDef
BSW Description		
M2 Template	**M2 Description**	
SystemTemplate	Arranges those signals that are transferred by the gateway from one channel to the other in pairs and defines the mapping between them.	
M2 Parameter		
Fibex::Fibex4Multiplatform::Gateway::SignalMapping		
Mapping Rule		**Mapping Type**

Create Container for each ISignalMapping that is defined in the ECU Extract.	full

BSW Module	BSW Context		
Com	Com/ComConfig/ComGwMapping		
BSW Parameter		**BSW Type**	
ComGwDestination		EcucChoiceContainerDef	
BSW Description			
Each instance of this choice container allows to define one routing destination either by reference to an already configured COM signal / signal group or by a destination description container.			
M2 Template	**M2 Description**		
SystemTemplate	Target destination of the referencing mapping.		
M2 Parameter			
Fibex::Fibex4Multiplatform::Gateway::SignalMapping.targetSignal			
Mapping Rule			**Mapping Type**
Create Container for each targetSignal reference that is defined in the ISignal Mapping.			full

BSW Module	BSW Context		
Com	Com/ComConfig/ComGwMapping		
BSW Parameter		**BSW Type**	
ComGwSource		EcucChoiceContainerDef	
BSW Description			
This choice container allows the definition of the gateway source signal either by reference to an already configured COM signal / signal group or by a source description container.			
M2 Template	**M2 Description**		
SystemTemplate	Source destination of the referencing mapping.		
M2 Parameter			
Fibex::Fibex4Multiplatform::Gateway::SignalMapping.sourceSignal			
Mapping Rule			**Mapping Type**
Create Container for sourceSignal reference that is defined in the ISignalMapping.			full

BSW Module	BSW Context		
Com	Com/ComConfig		
BSW Parameter		**BSW Type**	
ComIPdu		EcucParamConfContainerDef	
BSW Description			
Contains the configuration parameters of the AUTOSAR COM module's I-PDUs.			
M2 Template	**M2 Description**		
SystemTemplate	Represents the I-PDU's handled by Com. The SignalIPdu assembled and disassembled in AUTOSAR COM consists of one or more signals.		
M2 Parameter			
Fibex::FibexCore::CoreCommunication::SignalIPdu			
Mapping Rule			**Mapping Type**
create container for each SignalIPdu that is transmitted by the regarded ECU.			full

BSW Module	BSW Context	
Com	Com/ComConfig/ComIPdu	
BSW Parameter		**BSW Type**

ComIPduCallout		EcucFunctionNameDef	
BSW Description			
This parameter defines the existence and the name of a callout function for the corresponding I-PDU. If this parameter is omitted no I-PDU callout shall take place for the corresponding I-PDU.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	**BSW Context**		
Com	Com/ComConfig/ComIPdu		
BSW Parameter		**BSW Type**	
ComIPduCancellationSupport		EcucBooleanParamDef	
BSW Description			
Defines for I-PDUs with ComIPduType NORMAL: If the underlying IF-modul supports cancellation of transmit requests. Defines for I-PDUs with ComIPduType TP: If the underlying TP-module supports RX and TX cancellation of ongoing requests.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	**BSW Context**		
Com	Com/ComConfig/ComIPdu		
BSW Parameter		**BSW Type**	
ComIPduCounter		EcucParamConfContainerDef	
BSW Description			
This optional container contains the configuration parameters of PDU Counter.			
M2 Template	**M2 Description**		
SystemTemplate	A Pdu counter is included in a predefined set of PDUs and used to ensure that a sequence of PDUs is maintained.		
M2 Parameter			
Fibex::FibexCore::CoreCommunication::SignalIPdu.pduCounter			
Mapping Rule			**Mapping Type**
If pduCounter is aggregated by ISignalIPdu then create this container			full

BSW Module	**BSW Context**	
Com	Com/ComConfig/ComIPdu/ComIPduCounter	
BSW Parameter		**BSW Type**
ComIPduCounterErrorNotification		EcucFunctionNameDef
BSW Description		
Name of Com_CbkCounterErr callback function to be called. If this parameter is omitted no I-PDU counter mismatch notification shall take place.		
M2 Template	**M2 Description**	

M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	BSW Context	
Com	Com/ComConfig/ComIPdu/ComIPduCounter	
BSW Parameter		**BSW Type**
ComIPduCounterSize		EcucIntegerParamDef
BSW Description		
Size of PDU Counter expressed in bits		
M2 Template	**M2 Description**	
SystemTemplate	Size of PDU Counter expressed in bits.	
M2 Parameter		
Fibex::FibexCore::CoreCommunication::SignalIPduCounter.pduCounterSize		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
Com	Com/ComConfig/ComIPdu/ComIPduCounter	
BSW Parameter		**BSW Type**
ComIPduCounterStartPosition		EcucIntegerParamDef
BSW Description		
Position of PDU counter expressed in bits from start position of data content of I-PDU (SDU). Note that PDU counter is not allowed to cross a byte border. The parameter ComIPduCounterStartPosition shall define the bit0 of the first byte like in little endian byte order.		
M2 Template	**M2 Description**	
SystemTemplate	Position of PDU counter expressed in bits. Note that PDU counter is not allowed to cross a byte border.	
M2 Parameter		
Fibex::FibexCore::CoreCommunication::SignalIPduCounter.pduCounterStartPosition		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
Com	Com/ComConfig/ComIPdu/ComIPduCounter	
BSW Parameter		**BSW Type**
ComIPduCounterThreshold		EcucIntegerParamDef
BSW Description		
Threshold value of I-PDU counter algorithm, see COM590.		
M2 Template	**M2 Description**	
SystemTemplate	Threshold value of I-PDU counter algorithm. See AUTOSAR COM Spec for more details.	
M2 Parameter		
Fibex::FibexCore::CoreCommunication::SignalIPduCounter.pduCounterThreshold		
Mapping Rule		**Mapping Type**
1:1 mapping		full

— AUTOSAR CONFIDENTIAL —

<table>
<tr><th>BSW Module</th><th colspan="2">BSW Context</th></tr>
<tr><td>Com</td><td colspan="2">Com/ComConfig/ComIPdu</td></tr>
<tr><th colspan="2">BSW Parameter</th><th>BSW Type</th></tr>
<tr><td colspan="2">ComIPduDirection</td><td>EcucEnumerationParamDef</td></tr>
<tr><th colspan="3">BSW Description</th></tr>
<tr><td colspan="3">The direction defines if this I-PDU, and therefore the contributing signals and signal groups, shall be sent or received.</td></tr>
<tr><th>M2 Template</th><th colspan="2">M2 Description</th></tr>
<tr><td>SystemTemplate</td><td colspan="2">communication Direction of the Connector Port (input or output Port).</td></tr>
<tr><th colspan="3">M2 Parameter</th></tr>
<tr><td colspan="3">Fibex::FibexCore::CoreTopology::CommConnectorPort.communicationDirection</td></tr>
<tr><th colspan="2">Mapping Rule</th><th>Mapping Type</th></tr>
<tr><td colspan="2">Find IPduTriggering of the regarded SignalIPdu. The IPduTriggering contains a reference to an IPduPort that is aggegated by the regarded ECU. If the communicationDirection of the CommConnectorPort is "in" than the IPdu is received.</td><td>full</td></tr>
</table>

<table>
<tr><th>BSW Module</th><th colspan="2">BSW Context</th></tr>
<tr><td>Com</td><td colspan="2">Com/ComConfig/ComIPdu</td></tr>
<tr><th colspan="2">BSW Parameter</th><th>BSW Type</th></tr>
<tr><td colspan="2">ComIPduGroupRef</td><td>EcucReferenceDef</td></tr>
<tr><th colspan="3">BSW Description</th></tr>
<tr><td colspan="3">Reference to the I-PDU groups this I-PDU belongs to.</td></tr>
<tr><th>M2 Template</th><th colspan="2">M2 Description</th></tr>
<tr><td>SystemTemplate</td><td colspan="2">Reference to a set of SignalIPdus, which are contained in the I-Pdu Group.</td></tr>
<tr><th colspan="3">M2 Parameter</th></tr>
<tr><td colspan="3">Fibex::FibexCore::CoreCommunication::IPduGroup.ipdu</td></tr>
<tr><th colspan="2">Mapping Rule</th><th>Mapping Type</th></tr>
<tr><td colspan="2">Find IPduGroup that points to this SignalIPdu and create the reference.</td><td>full</td></tr>
</table>

<table>
<tr><th>BSW Module</th><th colspan="2">BSW Context</th></tr>
<tr><td>Com</td><td colspan="2">Com/ComConfig/ComIPdu</td></tr>
<tr><th colspan="2">BSW Parameter</th><th>BSW Type</th></tr>
<tr><td colspan="2">ComIPduHandleId</td><td>EcucIntegerParamDef</td></tr>
<tr><th colspan="3">BSW Description</th></tr>
<tr><td colspan="3">The numerical value used as the ID of this I-PDU. The ComIPduHandleId is required by the API calls to receive I-PDUs from the PduR (ComIP-duDirection: Receive). For Tx-I-PDUs (ComIPduDirection: Send) this handle Id is used by the PduR to confirm the transmission of the ComIPdu. In case no Tx-Confirmation is configured for a Tx-I-PDU, the ComIPduHandleId is not used.</td></tr>
<tr><th>M2 Template</th><th colspan="2">M2 Description</th></tr>
<tr><td></td><td colspan="2"></td></tr>
<tr><th colspan="3">M2 Parameter</th></tr>
<tr><td colspan="3"></td></tr>
<tr><th colspan="2">Mapping Rule</th><th>Mapping Type</th></tr>
<tr><td colspan="2"></td><td>local</td></tr>
</table>

<table>
<tr><th>BSW Module</th><th colspan="2">BSW Context</th></tr>
<tr><td>Com</td><td colspan="2">Com/ComConfig/ComIPdu</td></tr>
<tr><th colspan="2">BSW Parameter</th><th>BSW Type</th></tr>
<tr><td colspan="2">ComIPduReplication</td><td>EcucParamConfContainerDef</td></tr>
<tr><th colspan="3">BSW Description</th></tr>
</table>

This optional container contains the information needed for each I-PDU replicated.		
M2 Template	**M2 Description**	
SystemTemplate	PDU Replication is a form of redundancy where the data content of one PDU (source) is transmitted inside a set of replica PDUs.	
M2 Parameter		
Fibex::FibexCore::CoreCommunication::SignalIPdu.pduReplication		
Mapping Rule		**Mapping Type**
If pduReplication is defined for the SignalIPdu then create this container		full

BSW Module	**BSW Context**	
Com	Com/ComConfig/ComIPdu/ComIPduReplication	
BSW Parameter		**BSW Type**
ComIPduReplicaRef		EcucReferenceDef
BSW Description		
Reference to replicas PduR PDUs of this IPDU.		
M2 Template	**M2 Description**	
SystemTemplate	Reference to replica PDUs of this IPDU.	
M2 Parameter		
Fibex::FibexCore::CoreCommunication::SignalIPduReplication.replicaPdus		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
Com	Com/ComConfig/ComIPdu/ComIPduReplication	
BSW Parameter		**BSW Type**
ComIPduReplicationQuorum		EcucIntegerParamDef
BSW Description		
The number of identical I-PDUs needed for successful voting.		
M2 Template	**M2 Description**	
SystemTemplate	The number of identical I-PDUs needed for successful voting.	
M2 Parameter		
Fibex::FibexCore::CoreCommunication::SignalIPduReplication.pduReplicationVoting		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
Com	Com/ComConfig/ComIPdu	
BSW Parameter		**BSW Type**
ComIPduSignalGroupRef		EcucReferenceDef
BSW Description		
References to all signal groups contained in this I-Pdu		
M2 Template	**M2 Description**	
SystemTemplate	An ISignalToIPduMapping describes the mapping of ISignals to SignalIPdus and defines the position of the ISignal within an SignalIPdu.	
M2 Parameter		
Fibex::FibexCore::CoreCommunication::ISignalToIPduMapping		
Mapping Rule		**Mapping Type**
Find ISignal in the ISignalIPdu that refers to a ISignalGroup and create reference to this Group		full

BSW Module	**BSW Context**		
Com	Com/ComConfig/ComIPdu		
BSW Parameter		**BSW Type**	
ComIPduSignalProcessing		EcucEnumerationParamDef	
BSW Description			
For the definition of the two modes Immediate and Deferred.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	**BSW Context**		
Com	Com/ComConfig/ComIPdu		
BSW Parameter		**BSW Type**	
ComIPduSignalRef		EcucReferenceDef	
BSW Description			
References to all signals contained in this I-PDU.			
M2 Template	**M2 Description**		
SystemTemplate	An ISignalToIPduMapping describes the mapping of ISignals to SignalIPdus and defines the position of the ISignal within an SignalIPdu.		
M2 Parameter			
Fibex::FibexCore::CoreCommunication::ISignalToIPduMapping			
Mapping Rule			**Mapping Type**
Find ISignal in the IPdu which refers to a SystemSignal and create reference to this Signal.			full

BSW Module	**BSW Context**		
Com	Com/ComConfig/ComIPdu		
BSW Parameter		**BSW Type**	
ComIPduTriggerTransmitCallout		EcucFunctionNameDef	
BSW Description			
If there is a trigger transmit callout defined for this I-PDU this parameter contains the name of the callout function.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	**BSW Context**	
Com	Com/ComConfig/ComIPdu	
BSW Parameter		**BSW Type**
ComIPduType		EcucEnumerationParamDef
BSW Description		
Defines if this I-PDU is a normal I-PDU that can be sent unfragmented or if this is a large I-PDU that shall be sent via the Transport Protocol of the underlying bus.		
M2 Template	**M2 Description**	

SystemTemplate	TransportProtocols::CanTpConnection or TransportProtocols::FlexRayTpConnection or or TransportProtocols::LinTpConnection	
M2 Parameter		
A CanTpConnection represents an internal path for the transmission or reception of a Pdu via CanTp and describes the the sender and the receiver of this particular communication. The CanTp module routes a Pdu through the connection.		
Mapping Rule		**Mapping Type**
If this IPdu is mapped in the System Description by a TpConnection to NPdus than set this EnumerationLiteral to TP.		full

BSW Module	**BSW Context**	
Com	Com/ComConfig/ComIPdu	
BSW Parameter		**BSW Type**
ComPduIdRef		EcucReferenceDef
BSW Description		
Reference to the "global" Pdu structure to allow harmonization of handle IDs in the COM-Stack.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
Com	Com/ComConfig/ComIPdu	
BSW Parameter		**BSW Type**
ComTxIPdu		EcucParamConfContainerDef
BSW Description		
This container contains additional transmission related configuration parameters of the AUTOSAR COM module's I-PDUs.		
M2 Template	**M2 Description**	
SystemTemplate	Represents the IPdus handled by Com. The IPdu assembled and disassembled in AUTOSAR COM consists of one or more signals. In case no multiplexing is performed this IPdu is routed to/from the Interface Layer.	
M2 Parameter		
Fibex::FibexCore::CoreCommunication::ISignalIPdu		
Mapping Rule		**Mapping Type**
create container if an ISignalIPdu is transmitted by the regarded ECU.		full

BSW Module	**BSW Context**	
Com	Com/ComConfig/ComIPdu/ComTxIPdu	
BSW Parameter		**BSW Type**
ComMinimumDelayTime		EcucFloatParamDef
BSW Description		
Defines the Minimum Delay Time (MDT) between successive transmissions of this I-PDU in seconds. The MDT is independent of the possible different transmission modes. There is only one minimum delay time parameter for one I-PDU. The minimum delay timer is not reset by changing the transmission mode. Hence, it is not allowed to violate the minimum delay time by transmission mode changes. It is not possible to monitor the minimum delay time for I-PDUs that are requested using the Com_TriggerTransmit API.		
M2 Template	**M2 Description**	

SystemTemplate	Minimum Delay in seconds between successive transmissions of this I-PDU, independent of the Transmission Mode.	
M2 Parameter		
CoreCommunication::IPduTiming.minimumDelay		
Mapping Rule		**Mapping Type**
Find IPduTiming for the transmitted IPdu and use the specified value.		full

BSW Module	**BSW Context**	
Com	Com/ComConfig/ComIPdu/ComTxIPdu	
BSW Parameter		**BSW Type**
ComTxIPduClearUpdateBit		EcucEnumerationParamDef
BSW Description		
Defines when the update-bits of signals or signal groups, contained in this I-PDU, will be cleared.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
Com	Com/ComConfig/ComIPdu/ComTxIPdu	
BSW Parameter		**BSW Type**
ComTxIPduUnusedAreasDefault		EcucIntegerParamDef
BSW Description		
The AUTOSAR COM module fills not used areas of an I-PDU with this byte pattern. This attribute is mandatory to avoid undefined behaviour. This byte-pattern will be repeated throughout the I-PDU before any init-values or update-bits were set.		
M2 Template	**M2 Description**	
SystemTemplate	AUTOSAR COM fills not used areas of an IPDU with this bit-pattern. This attribute is mandatory to avoid undefined behavior. This byte-pattern will be repeated throughout the IPDU.	
M2 Parameter		
Fibex::FibexCore::CoreCommunication::IPdu.unusedBitPattern		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
Com	Com/ComConfig/ComIPdu/ComTxIPdu	
BSW Parameter		**BSW Type**
ComTxModeFalse		EcucParamConfContainerDef
BSW Description		
This container contains the configuration parameters of the AUTOSAR COM module's transmission modes in the case the ComFilter evaluates to false.		
M2 Template	**M2 Description**	
SystemTemplate	If the COM Transmission Mode is false the timing is aggregated by the TransmissionModeTiming element (role transmissionModeFalseTiming).	
M2 Parameter		
Fibex::FibexCore::CoreCommunication::SignalIPdu.iPduTimingSpecification		
Mapping Rule		**Mapping Type**
Create Container if a timing specification is defined for this IPdu.		full

<table>
<tr><th>BSW Module</th><th colspan="2">BSW Context</th></tr>
<tr><td>Com</td><td colspan="2">Com/ComConfig/ComIPdu/ComTxIPdu/ComTxModeFalse</td></tr>
<tr><th colspan="2">BSW Parameter</th><th>BSW Type</th></tr>
<tr><td colspan="2">ComTxMode</td><td>EcucParamConfContainerDef</td></tr>
<tr><th colspan="3">BSW Description</th></tr>
<tr><td colspan="3">This container contains the configuration parameters of the AUTOSAR COM module's transmission modes.</td></tr>
<tr><th>M2 Template</th><th colspan="2">M2 Description</th></tr>
<tr><td>SystemTemplate</td><td colspan="2">AUTOSAR COM provides the possibility to define two different TRANSMISSION MODES for each I-PDU. The Transmission Mode of an IPdu that is valid at a specific point in time is selected using the values of the signals that are mapped to this IPdu.</td></tr>
<tr><th colspan="3">M2 Parameter</th></tr>
<tr><td colspan="3">Fibex::FibexCore::CoreCommunication::SignalIPdu.iPduTimingSpecification</td></tr>
<tr><th colspan="2">Mapping Rule</th><th>Mapping Type</th></tr>
<tr><td colspan="2">Create Container if a timing specification is defined for this IPdu.</td><td>full</td></tr>
</table>

<table>
<tr><th>BSW Module</th><th colspan="2">BSW Context</th></tr>
<tr><td>Com</td><td colspan="2">Com/ComConfig/ComIPdu/ComTxIPdu/ComTxModeFalse/ComTxMode</td></tr>
<tr><th colspan="2">BSW Parameter</th><th>BSW Type</th></tr>
<tr><td colspan="2">ComTxModeMode</td><td>EcucEnumerationParamDef</td></tr>
<tr><th colspan="3">BSW Description</th></tr>
<tr><td colspan="3">The available transmission modes described in [18] shall be extended by the additional mode None.

The transmission mode None shall not have any further sub-attributes in the ComTxMode object.</td></tr>
<tr><th>M2 Template</th><th colspan="2">M2 Description</th></tr>
<tr><td>SystemTemplate</td><td colspan="2">If the COM Transmission Mode is false the timing is aggregated by the TransmitransmissionModeFalseTiming. If the COM Transmission Mode is true the timing is aggregated by the TransmissionModeTrueTiming.</td></tr>
<tr><th colspan="3">M2 Parameter</th></tr>
<tr><td colspan="3">Fibex::FibexCore::CoreCommunication::TransmissionModeTiming</td></tr>
<tr><th colspan="2">Mapping Rule</th><th>Mapping Type</th></tr>
<tr><td colspan="2">Periodic Mode is described by CyclicTiming. Direct /n-times Mode is described by EventControlledTiming. Mixed Mode is described if Cyclic and EventControlledTimings are assigned. None is described if no timing is assigned.</td><td>full</td></tr>
</table>

<table>
<tr><th>BSW Module</th><th colspan="2">BSW Context</th></tr>
<tr><td>Com</td><td colspan="2">Com/ComConfig/ComIPdu/ComTxIPdu/ComTxModeFalse/ComTxMode</td></tr>
<tr><th colspan="2">BSW Parameter</th><th>BSW Type</th></tr>
<tr><td colspan="2">ComTxModeNumberOfRepetitions</td><td>EcucIntegerParamDef</td></tr>
<tr><th colspan="3">BSW Description</th></tr>
<tr><td colspan="3">Defines the number of repetitions for the transmission mode DIRECT and the event driven part of transmission mode MIXED.</td></tr>
<tr><th>M2 Template</th><th colspan="2">M2 Description</th></tr>
<tr><td>SystemTemplate</td><td colspan="2">Defines the number of repetitions for the Direct/N-Times transmission mode and the event driven part of Mixed transmission mode.</td></tr>
<tr><th colspan="3">M2 Parameter</th></tr>
<tr><td colspan="3">Fibex::FibexCore::CoreCommunication.EventControlledTiming.numberOfRepetitions</td></tr>
<tr><th colspan="2">Mapping Rule</th><th>Mapping Type</th></tr>
</table>

If "EventControlledTiming.numberOfRepetitions" = 0 then ComTxModeNumberOfRepetitions = 0; If "EventControlledTiming.numberOfRepetitions" > 0 then ComTxModeNumberOfRepetitions = "EventControlledTiming.numberOfRepetitions" + 1	full

BSW Module	**BSW Context**	
Com	Com/ComConfig/ComIPdu/ComTxIPdu/ComTxModeFalse/ComTxMode	
BSW Parameter		**BSW Type**
ComTxModeRepetitionPeriod		EcucFloatParamDef
BSW Description		
Defines the repetition period in seconds of the multiple transmissions in case ComTxModeNumberOfRepetitions is configured greater than 1 and ComTxModeMode is configured to DIRECT or MIXED. In case of the mixed transmission mode only the event driven part is affected.		
M2 Template	**M2 Description**	
SystemTemplate	Specification of the time in seconds that elapses before the pdu can be sent the next time (Minimum repeat gap between two pdus)	
M2 Parameter		
Fibex::FibexCore::CoreCommunication::EventControlledTiming.repetitionPeriod		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
Com	Com/ComConfig/ComIPdu/ComTxIPdu/ComTxModeFalse/ComTxMode	
BSW Parameter		**BSW Type**
ComTxModeTimeOffset		EcucFloatParamDef
BSW Description		
Defines the period in seconds between the start of the I-PDU by Com_IpduGroupControl and the first transmission request in case ComTxModeMode is configured to PERIODIC or MIXED. In case of the mixed transmission mode only the periodic part is affected.		
M2 Template	**M2 Description**	
SystemTemplate	Specification of the time that is needed before the pdu can be sent the first time.	
M2 Parameter		
Fibex::FibexCore::CoreCommunication::CyclicTiming.StartingTime		
Mapping Rule		**Mapping Type**
The value for the True and the False Transmission Mode can be derived from IPduTiming.TransmissionModeDeclaration.TransmissionModeTiming element		full

BSW Module	**BSW Context**	
Com	Com/ComConfig/ComIPdu/ComTxIPdu/ComTxModeFalse/ComTxMode	
BSW Parameter		**BSW Type**
ComTxModeTimePeriod		EcucFloatParamDef
BSW Description		
Defines the repetition period in seconds of the periodic transmission requests in case ComTxModeMode is configured to PERIODIC or MIXED. In case of the mixed transmission mode only the periodic part is affected.		
M2 Template	**M2 Description**	
SystemTemplate	Period of the repetition of cyclic transmissions.	
M2 Parameter		
Fibex::FibexCore::CoreCommunication::CyclicTiming.timePeriod		
Mapping Rule		**Mapping Type**

The value for the True and the False Transmission Mode can be derived from I PduTiming.TransmissionModeDeclaration.TransmissionModeTiming element	full

BSW Module	**BSW Context**	
Com	Com/ComConfig/ComIPdu/ComTxIPdu	
BSW Parameter		**BSW Type**
ComTxModeTrue		EcucParamConfContainerDef
BSW Description		
This container contains the configuration parameters of the AUTOSAR COM module's transmission modes in the case the ComFilter evaluates to true.		
M2 Template	**M2 Description**	
SystemTemplate	If the COM Transmission Mode is true the timing can be aggregated by the TransmissionModeTiming element (role transmissionModeTrueTiming)	
M2 Parameter		
Fibex::FibexCore::CoreCommunication::SignalIPdu.iPduTimingSpecification		
Mapping Rule		**Mapping Type**
Create Container if a timing specification is defined for this IPdu.		full

BSW Module	**BSW Context**	
Com	Com/ComConfig/ComIPdu/ComTxIPdu/ComTxModeTrue	
BSW Parameter		**BSW Type**
ComTxMode		EcucParamConfContainerDef
BSW Description		
This container contains the configuration parameters of the AUTOSAR COM module's transmission modes.		
M2 Template	**M2 Description**	
SystemTemplate	AUTOSAR COM provides the possibility to define two different TRANSMISSION MODES for each I-PDU. The Transmission Mode of an IPdu that is valid at a specific point in time is selected using the values of the signals that are mapped to this IPdu.	
M2 Parameter		
Fibex::FibexCore::CoreCommunication::SignalIPdu.iPduTimingSpecification		
Mapping Rule		**Mapping Type**
Create Container if a timing specification is defined for this IPdu.		full

BSW Module	**BSW Context**	
Com	Com/ComConfig/ComIPdu/ComTxIPdu/ComTxModeTrue/ComTxMode	
BSW Parameter		**BSW Type**
ComTxModeMode		EcucEnumerationParamDef
BSW Description		
The available transmission modes described in [18] shall be extended by the additional mode None. The transmission mode None shall not have any further sub-attributes in the ComTxMode object.		
M2 Template	**M2 Description**	
SystemTemplate	If the COM Transmission Mode is false the timing is aggregated by the Trans-mitransmissionModeFalseTiming. If the COM Transmission Mode is true the timing is aggregated by the TransmissionModeTrueTiming.	
M2 Parameter		
Fibex::FibexCore::CoreCommunication::TransmissionModeTiming		
Mapping Rule		**Mapping Type**

— AUTOSAR CONFIDENTIAL —

Periodic Mode is described by CyclicTiming. Direct /n-times Mode is described by EventControlledTiming. Mixed Mode is described if Cyclic and EventControlledTimings are assigned. None is described if no timing is assigned.	full

BSW Module	**BSW Context**	
Com	Com/ComConfig/ComIPdu/ComTxIPdu/ComTxModeTrue/ComTxMode	
BSW Parameter		**BSW Type**
ComTxModeNumberOfRepetitions		EcucIntegerParamDef
BSW Description		
Defines the number of repetitions for the transmission mode DIRECT and the event driven part of transmission mode MIXED.		
M2 Template	**M2 Description**	
SystemTemplate	Defines the number of repetitions for the Direct/N-Times transmission mode and the event driven part of Mixed transmission mode.	
M2 Parameter		
Fibex::FibexCore::CoreCommunication.EventControlledTiming.numberOfRepetitions		
Mapping Rule		**Mapping Type**
If "EventControlledTiming.numberOfRepetitions" = 0 then ComTxModeNumberOfRepetitions = 0; If "EventControlledTiming.numberOfRepetitions" > 0 then ComTxModeNumberOfRepetitions = "EventControlledTiming.numberOfRepetitions" + 1		full

BSW Module	**BSW Context**	
Com	Com/ComConfig/ComIPdu/ComTxIPdu/ComTxModeTrue/ComTxMode	
BSW Parameter		**BSW Type**
ComTxModeRepetitionPeriod		EcucFloatParamDef
BSW Description		
Defines the repetition period in seconds of the multiple transmissions in case ComTxModeNumberOfRepetitions is configured greater than 1 and ComTxModeMode is configured to DIRECT or MIXED. In case of the mixed transmission mode only the event driven part is affected.		
M2 Template	**M2 Description**	
SystemTemplate	Specification of the time in seconds that elapses before the pdu can be sent the next time (Minimum repeat gap between two pdus)	
M2 Parameter		
Fibex::FibexCore::CoreCommunication::EventControlledTiming.repetitionPeriod		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
Com	Com/ComConfig/ComIPdu/ComTxIPdu/ComTxModeTrue/ComTxMode	
BSW Parameter		**BSW Type**
ComTxModeTimeOffset		EcucFloatParamDef
BSW Description		
Defines the period in seconds between the start of the I-PDU by Com_IpduGroupControl and the first transmission request in case ComTxModeMode is configured to PERIODIC or MIXED. In case of the mixed transmission mode only the periodic part is affected.		
M2 Template	**M2 Description**	
SystemTemplate	Specification of the time that is needed before the pdu can be sent the first time.	
M2 Parameter		
Fibex::FibexCore::CoreCommunication::CyclicTiming.StartingTime		
Mapping Rule		**Mapping Type**

The value for the True and the False Transmission Mode can be derived from IPduTiming.TransmissionModeDeclaration.TransmissionModeTiming element	full

BSW Module	BSW Context	
Com	Com/ComConfig/ComIPdu/ComTxIPdu/ComTxModeTrue/ComTxMode	
BSW Parameter		**BSW Type**
ComTxModeTimePeriod		EcucFloatParamDef
BSW Description		
Defines the repetition period in seconds of the periodic transmission requests in case ComTxMode-Mode is configured to PERIODIC or MIXED. In case of the mixed transmission mode only the periodic part is affected.		
M2 Template	**M2 Description**	
SystemTemplate	Period of the repetition of cyclic transmissions.	
M2 Parameter		
Fibex::FibexCore::CoreCommunication::CyclicTiming.timePeriod		
Mapping Rule		**Mapping Type**
The value for the True and the False Transmission Mode can be derived from IPduTiming.TransmissionModeDeclaration.TransmissionModeTiming element		full

BSW Module	BSW Context	
Com	Com/ComConfig	
BSW Parameter		**BSW Type**
ComIPduGroup		EcucParamConfContainerDef
BSW Description		
Contains the configuration parameters of the AUTOSAR COM module's I-PDU groups.		
M2 Template	**M2 Description**	
SystemTemplate	The AUTOSAR COM Layer is able to start and to stop sending and receiving configurable groups of I-Pdus during runtime. An I-Pdu group contains either Com I-Pdus or I-Pdu groups.	
M2 Parameter		
Fibex::FibexCore::CoreCommunication::ISignalIPduGroup		
Mapping Rule		**Mapping Type**
Create container for each CoreCommunication::ISignalIPduGroup that is contained in the ECU Extract.		full

BSW Module	BSW Context	
Com	Com/ComConfig/ComIPduGroup	
BSW Parameter		**BSW Type**
ComIPduGroupGroupRef		EcucReferenceDef
BSW Description		
References to all I-PDU groups that includes this I-PDU group. If this reference is omitted this I-PDU group does not belong to another I-PDU group.		
M2 Template	**M2 Description**	
SystemTemplate	An I-PDU group can be included in other I-Pdu groups.	
M2 Parameter		
CoreCommunication::IPduGroup.containedIPduGroup		
Mapping Rule		**Mapping Type**
If the IPduGroup has a reference to a contained IPduGroup then create this reference.		full

BSW Module	BSW Context	
Com	Com/ComConfig/ComIPduGroup	
BSW Parameter		**BSW Type**
ComIPduGroupHandleId		EcucIntegerParamDef
BSW Description		
The numerical value used as the ID of this I-PDU Group . The ComIPduGroupHandleId is required by the API calls to start and stop I-PDU Groups. Range: 0 .. (ComSupportedIPduGroups-1)		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
Com	Com/ComConfig	
BSW Parameter		**BSW Type**
ComSignal		EcucParamConfContainerDef
BSW Description		
Contains the configuration parameters of the AUTOSAR COM module's signals.		
M2 Template	**M2 Description**	
SystemTemplate	An ISignalToIPduMapping describes the mapping of ISignals to SignalIPdus and defines the position of the ISignal within an SignalIPdu.	
M2 Parameter		
Fibex::FibexCore::CoreCommunication::ISignalToPduMapping		
Mapping Rule		**Mapping Type**
A Com signal must be defined for each ISignalToPduMapping that is transmitted or received by the regarded ECU.		full

BSW Module	BSW Context	
Com	Com/ComConfig/ComSignal	
BSW Parameter		**BSW Type**
ComBitPosition		EcucIntegerParamDef
BSW Description		
Starting position within the I-PDU. This parameter refers to the position in the I-PDU and not in the shadow buffer. If the endianness conversion is configured to Opaque the parameter ComBitPosition shall define the bit0 of the first byte like in little endian byte order		
M2 Template	**M2 Description**	
SystemTemplate	This parameter is necessary to describe the bitposition of a signal within an SignalIPdu.	
M2 Parameter		
Fibex::FibexCore::CoreCommunication::ISignalToIPduMapping.startPostion		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
Com	Com/ComConfig/ComSignal	
BSW Parameter		**BSW Type**

ComBitSize		EcucIntegerParamDef
BSW Description		
Size in bits, for non-array signal types. For ComSignalType UINT8_N and UINT8_DYN this size shall be configured by ComSignalLength.		
M2 Template	**M2 Description**	
SystemTemplate	Size of the signal in bits.	
M2 Parameter		
Fibex::FibexCore::CoreCommunication::SystemSignal.length		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
Com	Com/ComConfig/ComSignal	
BSW Parameter		**BSW Type**
ComDataInvalidAction		EcucEnumerationParamDef
BSW Description		
This parameter defines the action performed upon reception of an invalid signal. Relating to signal groups the action in case if one of the included signals is an invalid signal. If Replace is used the ComSignalInitValue will be used for the replacement.		
M2 Template	**M2 Description**	
SWComponentTemplate	Specifies whether the component can actively invalidate a particular dataElement.	
M2 Parameter		
PortInterface::SenderReceiverInterface.invalidationPolicy		
Mapping Rule		**Mapping Type**
If strategy keep is defined than set parameter to notify. If strategy replace is defined than set parameter to replace.		full

BSW Module	**BSW Context**	
Com	Com/ComConfig/ComSignal	
BSW Parameter		**BSW Type**
ComErrorNotification		EcucFunctionNameDef
BSW Description		
Only valid on sender side: Name of Com_CbkTxErr callback function to be called. If this parameter is omitted no error notification shall take place.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
Com	Com/ComConfig/ComSignal	
BSW Parameter		**BSW Type**
ComFilter		EcucParamConfContainerDef
BSW Description		
This container contains the configuration parameters of the AUTOSAR COM module's Filters. Note: On sender side the container is used to specify the transmission mode conditions.		

M2 Template	M2 Description	
SWComponentTemplate and SystemTemplate	Base class for data filters. The type of the filter is specified in dataFilterAttribute. Some of the filter types require additional arguments which are specified as attributes of this class.	
M2 Parameter		
SWCTemplate::Communication::ReceiverComSpec.filter (receiver side) and SystemTemplate::Fibex::FibexCore::CoreCommunication::IPduTiming::TransmissionModeDeclaration::TransmissionModeCondition::DataFilter		
Mapping Rule		**Mapping Type**
Create container on the receiver side if the ReceiverComSpec contains a Data Filter. Create Container on the sender side if the TransmissionModeCondition element contains a reference to this signal.		full

BSW Module	BSW Context		
Com	Com/ComConfig/ComSignal/ComFilter		
BSW Parameter		**BSW Type**	
ComFilterAlgorithm		EcucEnumerationParamDef	
BSW Description			
The range of values is specified in the [17] specification, chapter 2.2.2, Reception Filtering.			
M2 Template	**M2 Description**		
SWComponentTemplate and SystemTemplate	This attriburte specifies the type of the filter.		
M2 Parameter			
CommonStructure::DataFilter.dataFilterType			
Mapping Rule			**Mapping Type**
Mapping between DataFilterTypeEnum and ComFilterAlgorithm Enum is necessary.			full

BSW Module	BSW Context		
Com	Com/ComConfig/ComSignal/ComFilter		
BSW Parameter		**BSW Type**	
ComFilterMask		EcucIntegerParamDef	
BSW Description			
The name of this attribute corresponds to the parameter name in the [17] specification of Reception Filtering.			
M2 Template	**M2 Description**		
SWComponentTemplate and SystemTemplate	mask for old and new value		
M2 Parameter			
CommonStructure::DataFilter.mask			
Mapping Rule			**Mapping Type**
1:1 mapping			full

BSW Module	BSW Context	
Com	Com/ComConfig/ComSignal/ComFilter	
BSW Parameter		**BSW Type**
ComFilterMax		EcucIntegerParamDef
BSW Description		

The name of this attribute corresponds to the parameter name in the [17] specification of Reception Filtering.		
M2 Template	**M2 Description**	
SWComponentTemplate and SystemTemplate	Value to specify the upper boundary	
M2 Parameter		
CommonStructure::DataFilter::max		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
Com	Com/ComConfig/ComSignal/ComFilter	
BSW Parameter		**BSW Type**
ComFilterMin		EcucIntegerParamDef
BSW Description		
The name of this attribute corresponds to the parameter name in the [17] specification of Reception Filtering.		
M2 Template	**M2 Description**	
SystemTemplate and SWComponentTemplate	Value to specify the lower boundary	
M2 Parameter		
CommonStruture::DataFilter.min		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
Com	Com/ComConfig/ComSignal/ComFilter	
BSW Parameter		**BSW Type**
ComFilterOffset		EcucIntegerParamDef
BSW Description		
The name of this attribute corresponds to the parameter name in the [17] specification of Reception Filtering. Range = 0..(ComFilterPeriod-1)		
M2 Template	**M2 Description**	
SWComponentTemplate and SystemTemplate	Specifies the initial number of messages to occur before the first message is passed	
M2 Parameter		
CommonStructure::DataFilter.offset		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	**BSW Context**	
Com	Com/ComConfig/ComSignal/ComFilter	
BSW Parameter		**BSW Type**
ComFilterPeriod		EcucIntegerParamDef
BSW Description		
This parameter defines the period of the ComFilterAlgorithm ONE_EVERY_N.		

M2 Template	M2 Description	
SWComponentTemplate and SystemTemplate	specifies number of messages to occur before the message is passed again	
M2 Parameter		
CommonStructure::DataFilter::period		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
Com	Com/ComConfig/ComSignal/ComFilter	
BSW Parameter		**BSW Type**
ComFilterX		EcucIntegerParamDef
BSW Description		
The name of this attribute corresponds to the parameter name in the [17] specification of Reception Filtering.		
M2 Template	**M2 Description**	
SystemTemplate and SWComponentTemplate	Value to compare with	
M2 Parameter		
CommonStructure::DataFilter.x		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
Com	Com/ComConfig/ComSignal	
BSW Parameter		**BSW Type**
ComFirstTimeout		EcucFloatParamDef
BSW Description		
Defines the length of the first deadline monitoring timeout period in seconds. This timeout is used immediately after start (or restart) of the deadline monitoring service. The timeout period of the successive periods is configured by COM263_Conf.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
Com	Com/ComConfig/ComSignal	
BSW Parameter		**BSW Type**
ComHandleId		EcucIntegerParamDef
BSW Description		
The numerical value used as the ID. For signals it is required by the API calls Com_UpdateShadowSignal, Com_ReceiveShadowSignal and Com_InvalidateShadowSignal. For signals groups it is required by the Com_SendSignalGroup and Com_ReceiveSignalGroup calls.		

<table>
<tr><th>M2 Template</th><th colspan="2">M2 Description</th></tr>
<tr><td></td><td colspan="2"></td></tr>
<tr><th colspan="3">M2 Parameter</th></tr>
<tr><td colspan="3"></td></tr>
<tr><th colspan="2">Mapping Rule</th><th>Mapping Type</th></tr>
<tr><td colspan="2"></td><td>local</td></tr>
</table>

<table>
<tr><th>BSW Module</th><th colspan="3">BSW Context</th></tr>
<tr><td>Com</td><td colspan="3">Com/ComConfig/ComSignal</td></tr>
<tr><th colspan="2">BSW Parameter</th><th colspan="2">BSW Type</th></tr>
<tr><td colspan="2">ComInvalidNotification</td><td colspan="2">EcucFunctionNameDef</td></tr>
<tr><th colspan="4">BSW Description</th></tr>
<tr><td colspan="4">Only valid on receiver side: Name of Com_CbkInv callback function to be called. Name of the function which notifies the RTE about the reception of an invalidated signal/ signal group. Only applicable if ComDataInvalidAction is configured to NOTIFY.</td></tr>
<tr><th>M2 Template</th><th colspan="3">M2 Description</th></tr>
<tr><td></td><td colspan="3"></td></tr>
<tr><th colspan="4">M2 Parameter</th></tr>
<tr><td colspan="4"></td></tr>
<tr><th colspan="3">Mapping Rule</th><th>Mapping Type</th></tr>
<tr><td colspan="3"></td><td>local</td></tr>
</table>

<table>
<tr><th>BSW Module</th><th colspan="3">BSW Context</th></tr>
<tr><td>Com</td><td colspan="3">Com/ComConfig/ComSignal</td></tr>
<tr><th colspan="2">BSW Parameter</th><th colspan="2">BSW Type</th></tr>
<tr><td colspan="2">ComNotification</td><td colspan="2">EcucFunctionNameDef</td></tr>
<tr><th colspan="4">BSW Description</th></tr>
<tr><td colspan="4">On sender side: Name of Com_CbkTxAck callback function to be called.
On receiver side: Name of Com_CbkRxAck callback function to be called.

If this parameter is omitted no notification shall take place.</td></tr>
<tr><th>M2 Template</th><th colspan="3">M2 Description</th></tr>
<tr><td></td><td colspan="3"></td></tr>
<tr><th colspan="4">M2 Parameter</th></tr>
<tr><td colspan="4"></td></tr>
<tr><th colspan="3">Mapping Rule</th><th>Mapping Type</th></tr>
<tr><td colspan="3"></td><td>local</td></tr>
</table>

<table>
<tr><th>BSW Module</th><th colspan="2">BSW Context</th></tr>
<tr><td>Com</td><td colspan="2">Com/ComConfig/ComSignal</td></tr>
<tr><th colspan="2">BSW Parameter</th><th>BSW Type</th></tr>
<tr><td colspan="2">ComRxDataTimeoutAction</td><td>EcucEnumerationParamDef</td></tr>
<tr><th colspan="3">BSW Description</th></tr>
<tr><td colspan="3">This parameter defines the action performed upon expiration of the reception deadline monitoring timer.</td></tr>
<tr><th>M2 Template</th><th colspan="2">M2 Description</th></tr>
<tr><td>SWComponentTemplate</td><td colspan="2">Strategies of handling a reception timeout violation.</td></tr>
<tr><th colspan="3">M2 Parameter</th></tr>
<tr><td colspan="3">SWCTemplate::Communication:ReceiverComSpec::NonqueuedReceiverComSpec.handleTimeoutType</td></tr>
</table>

Mapping Rule	Mapping Type
1:1 mapping	full

BSW Module	BSW Context	
Com	Com/ComConfig/ComSignal	
BSW Parameter		**BSW Type**
ComSignalDataInvalidValue		EcucStringParamDef
BSW Description		
Defines the data invalid value of the signal. In case the ComSignalType is UINT8, UINT16, UINT32, SINT8, SINT16, SINT32 the string shall be interpreted as defined in the chapter Integer Type in the AUTOSAR EcuC specification. In case the ComSignalType is FLOAT32, FLOAT64 the string shall be interpreted as defined in the chapter Float Type in the AUTOSAR EcuC specification. In case the ComSignalType is BOOLEAN the string shall be interpreted as defined in the chapter Boolean Type in the AUTOSAR EcuC specification. In case the ComSignal is a UINT8_N, UINT6_DYN the string shall be interpreted as a decimal representation of the characters separated by blanks, e.g. "97 98 100" means a string "abd", where the char "a" is in byte 0(lowest address), "b" is in byte 1, and "d" is in byte 2 and (highest address).		
M2 Template	**M2 Description**	
SystemTemplate	Optional value to express invalidity of the actual data element.	
M2 Parameter		
SystemTemplate::CoreCommunication::ISignal::swDataDefProps.invalidValue		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
Com	Com/ComConfig/ComSignal	
BSW Parameter		**BSW Type**
ComSignalEndianness		EcucEnumerationParamDef
BSW Description		
Defines the endianness of the signal's network representation.		
M2 Template	**M2 Description**	
SystemTemplate	This parameter defines the order of the bytes of the signal and the packing into the IPdu. The byte ordering Little Endian (MostSignificantByteLast), Big Endian (MostSignificantByteFirst) and Opaque can be selected.	
M2 Parameter		
Fibex::FibexCore::CoreCommunication::ISignalToIPduMapping.packingByteOrder		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
Com	Com/ComConfig/ComSignal	
BSW Parameter		**BSW Type**
ComSignalInitValue		EcucStringParamDef
BSW Description		

Initial value for this signal. The default value is 0. In case the ComSignalType is UINT8, UINT16, UINT32, SINT8, SINT16, SINT32 the string shall be interpreted as defined in the chapter Integer Type in the AUTOSAR EcuC specification. In case the ComSignalType is FLOAT32, FLOAT64 the string shall be interpreted as defined in the chapter Float Type in the AUTOSAR EcuC specification. In case the ComSignalType is BOOLEAN the string shall be interpreted as defined in the chapter Boolean Type in the AUTOSAR EcuC specification. In case the ComSignal is a UINT8_N, UINT6_DYN the string shall be interpreted as a decimal representation of the characters separated by blanks, e.g. "97 98 100" means a string "abd", where the char "a" is in byte 0(lowest address), "b" is in byte 1, and "d" is in byte 2 and (highest address).	
M2 Template	**M2 Description**
SystemTemplate and SWComponentTemplate	If a full DataMapping exist this information may be available from a configured SenderComSpec and ReceiverComSpec. In case the System Description doesn't use a complete Software ComponentDescription an optional reference from SystemSignal is used.
M2 Parameter	
SWComponentTemplate::Communcation::ComSpec.initValue or SystemTemplate::Fibex::Fibex Core::CoreCommuncation::ISignal.initValue;	
Mapping Rule	**Mapping Type**
It is possible to aggregate an initValue at the level of a ComSpec in the SW C Template. in case the System Description doesn't use a complete Software Component Description (VFB View) the initValue is defined in the System Template.	full

BSW Module	**BSW Context**
Com	Com/ComConfig/ComSignal
BSW Parameter	**BSW Type**
ComSignalLength	EcucIntegerParamDef
BSW Description	
Description: For ComSignalType UINT8_N this parameter specifies the length n in bytes. For ComSignalType UINT8_DYN it specifies the maximum length in bytes. For all other types this parameter shall be ignored. Range: 0..8 for normal CAN/ LIN I-PDUs, 0..254 for normal FlexRay I-PDUs, and 0..4095 for I-PDUs with ComIPduType TP.	
M2 Template	**M2 Description**
SWComponentTemplate	The number of bits that are used to make up the opaque type.
M2 Parameter	
Datatype::DataTypes::OpaqueType.numberOfBits	
Mapping Rule	**Mapping Type**
Opaque data shall always be of uint8[n] and shall always be mapped to an n-bytes sized signal.	full

BSW Module	**BSW Context**
Com	Com/ComConfig/ComSignal
BSW Parameter	**BSW Type**
ComSignalType	EcucEnumerationParamDef
BSW Description	

The AUTOSAR type of the signal. Whether or not the signal is signed or unsigned can be found by examining the value of this attribute. This type could also be used to reserved appropriate storage in AUTOSAR COM.		
M2 Template	**M2 Description**	
SystemTemplate	Describes a reusable data type on the implementation level. This will typically correspond to a typedef in C-code.	
M2 Parameter		
SystemTemplate::Fibex::FibexCore::CoreCommunication::SystemSignal.networkRepresentationProps.swBaseType		
Mapping Rule		**Mapping Type**
Mapping of AUTOSAR data types (defined in the software component description) to COM Signal Types. Mapping rules are described in System Template Specification.		full

BSW Module	**BSW Context**	
Com	Com/ComConfig/ComSignal	
BSW Parameter		**BSW Type**
ComSystemTemplateSystemSignalRef		EcucForeignReferenceDef
BSW Description		
Reference to the ISignalToIPduMapping that contains a reference to the ISignal (System Template) which this ComSignal (or ComGroupSignal) represents.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**

BSW Module	**BSW Context**	
Com	Com/ComConfig/ComSignal	
BSW Parameter		**BSW Type**
ComTimeout		EcucFloatParamDef
BSW Description		
Defines the length of the deadline monitoring timeout period in seconds. The period for the first timeout period can be configured separately by COM183_Conf.		
M2 Template	**M2 Description**	
SWComponentTemplate or SystemTemplate	Timeout value in seconds for the reception of the ISignal.	
M2 Parameter		
SWCTemplate::Communication::ReceiverComSpec.aliveTimeout and SystemTemplate::Fibex::FibexCore::CoreCommunication::SignalPort.timeout		
Mapping Rule		**Mapping Type**
If a full DataMapping exist for the SystemSignal this information may be available from a configured ReceiverComSpec, in this case the timeout value in ReceiverComSpec override the optional timeout specification in the System Template.		full

BSW Module	**BSW Context**	
Com	Com/ComConfig/ComSignal	
BSW Parameter		**BSW Type**
ComTimeoutNotification		EcucFunctionNameDef

BSW Description		
On sender side: Name of Com_CbkTxTOut callback function to be called. On receiver side: Name of Com_CbkRxTOut callback function to be called.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context		
Com	Com/ComConfig/ComSignal		
BSW Parameter		**BSW Type**	
ComTransferProperty		EcucEnumerationParamDef	
BSW Description			
Defines if a write access to this signal can trigger the transmission of the corresponding I-PDU. If the I-PDU is triggered, depends also on the transmission mode of the corresponding I-PDU.			
M2 Template	**M2 Description**		
SystemTemplate	The triggered transfer property causes immediate transmission of the IPdu, except if transmission mode Periodic or transmission mode NONE is defined for the IPdu. The Pending transfer property does not cause transmission of an I-PDU.		
M2 Parameter			
Fibex::FibexCore::CoreCommunication::ISignalToIPduMapping.transferProperty			
Mapping Rule			**Mapping Type**
1:1 mapping			full

BSW Module	BSW Context		
Com	Com/ComConfig/ComSignal		
BSW Parameter		**BSW Type**	
ComUpdateBitPosition		EcucIntegerParamDef	
BSW Description			
Bit position of update-bit inside I-PDU. If this attribute is omitted then there is no update-bit. This setting must be consistently on sender and on receiver side. Range: 0..63 for CAN and LIN 0..2031 for FlexRay			
M2 Template	**M2 Description**		
SystemTemplate	The UpdateIndicationBit indicates to the receivers that the signal (or the signal group) was updated by the sender. Length is always one bit. The UpdateIndicationBitPosition attribute describes the position of the update bit within the SignalIPdu.		
M2 Parameter			
Fibex::FibexCore::CoreCommunication::ISignalToIPduMapping.updateIndicationBitPosition			
Mapping Rule			**Mapping Type**
1:1 mapping			full

BSW Module	BSW Context	
Com	Com/ComConfig	
BSW Parameter		**BSW Type**

ComSignalGroup		EcucParamConfContainerDef	
BSW Description			
Contains the configuration parameters of the AUTOSAR COM module's signal groups.			
M2 Template	**M2 Description**		
SystemTemplate	An ISignalGroup refers to a set of ISignals that must always be kept together. A ISignalGroup represents a COM Signal Group.		
M2 Parameter			
Fibex::FibexCore::CoreCommunication::ISignalGroup			
Mapping Rule			**Mapping Type**
Create this container for each ISignalGroup that exist in the ECU Extract.			full

BSW Module	**BSW Context**		
Com	Com/ComConfig/ComSignalGroup		
BSW Parameter		**BSW Type**	
ComDataInvalidAction		EcucEnumerationParamDef	
BSW Description			
This parameter defines the action performed upon reception of an invalid signal. Relating to signal groups the action in case if one of the included signals is an invalid signal. If Replace is used the ComSignalInitValue will be used for the replacement.			
M2 Template	**M2 Description**		
SWComponentTemplate	Specifies whether the component can actively invalidate a particular dataElement.		
M2 Parameter			
PortInterface::SenderReceiverInterface.invalidationPolicy			
Mapping Rule			**Mapping Type**
If strategy keep is defined than set parameter to notify. If strategy replace is defined than set parameter to replace.			full

BSW Module	**BSW Context**		
Com	Com/ComConfig/ComSignalGroup		
BSW Parameter		**BSW Type**	
ComErrorNotification		EcucFunctionNameDef	
BSW Description			
Only valid on sender side: Name of Com_CbkTxErr callback function to be called. If this parameter is omitted no error notification shall take place.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	**BSW Context**		
Com	Com/ComConfig/ComSignalGroup		
BSW Parameter		**BSW Type**	
ComFirstTimeout		EcucFloatParamDef	
BSW Description			
Defines the length of the first deadline monitoring timeout period in seconds. This timeout is used immediately after start (or restart) of the deadline monitoring service. The timeout period of the successive periods is configured by COM263_Conf.			

M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**
Com	Com/ComConfig/ComSignalGroup
BSW Parameter	**BSW Type**
ComGroupSignal	EcucParamConfContainerDef
BSW Description	
This container contains the configuration parameters of group signals. I.e. signals that are included within a signal group.	
M2 Template	**M2 Description**
SystemTemplate	The SystemSignalGroup element in the System Description contains a reference to a set of signals that must always be kept together.
M2 Parameter	
Fibex::FibexCore::CoreCommunication::SystemSignal	
Mapping Rule	**Mapping Type**
Create Container for each ISignal that is contained in the ISignalGroup.	full

BSW Module	**BSW Context**
Com	Com/ComConfig/ComSignalGroup/ComGroupSignal
BSW Parameter	**BSW Type**
ComBitPosition	EcucIntegerParamDef
BSW Description	
Starting position within the I-PDU. This parameter refers to the position in the I-PDU and not in the shadow buffer. If the endianness conversion is configured to Opaque the parameter ComBitPosition shall define the bit0 of the first byte like in little endian byte order	
M2 Template	**M2 Description**
SystemTemplate	This parameter is necessary to describe the bitposition of a signal within an SignalIPdu.
M2 Parameter	
Fibex::FibexCore::CoreCommunication::ISignalToIPduMapping.startPostion	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	**BSW Context**
Com	Com/ComConfig/ComSignalGroup/ComGroupSignal
BSW Parameter	**BSW Type**
ComBitSize	EcucIntegerParamDef
BSW Description	
Size in bits, for non-array signal types. For ComSignalType UINT8_N and UINT8_DYN this size shall be configured by ComSignalLength.	
M2 Template	**M2 Description**
SystemTemplate	Size of the signal in bits.
M2 Parameter	
Fibex::FibexCore::CoreCommunication::SystemSignal.length	
Mapping Rule	**Mapping Type**
1:1 mapping	full

<table>
<tr><th>BSW Module</th><th colspan="3">BSW Context</th></tr>
<tr><td>Com</td><td colspan="3">Com/ComConfig/ComSignalGroup/ComGroupSignal</td></tr>
<tr><th colspan="2">BSW Parameter</th><th colspan="2">BSW Type</th></tr>
<tr><td colspan="2">ComFilter</td><td colspan="2">EcucParamConfContainerDef</td></tr>
<tr><th colspan="4">BSW Description</th></tr>
<tr><td colspan="4">This container contains the configuration parameters of the AUTOSAR COM module's Filters.

Note: On sender side the container is used to specify the transmission mode conditions.</td></tr>
<tr><th>M2 Template</th><th colspan="3">M2 Description</th></tr>
<tr><td>SWComponentTemplate and SystemTemplate</td><td colspan="3">Base class for data filters. The type of the filter is specified in dataFilterAttribute. Some of the filter types require additional arguments which are specified as attributes of this class.</td></tr>
<tr><th colspan="4">M2 Parameter</th></tr>
<tr><td colspan="4">SWCTemplate::Communication::ReceiverComSpec.filter (receiver side) and SystemTemplate::Fibex::FibexCore::CoreCommunication::IPduTiming::TransmissionModeDeclaration::TransmissionModeCondition::DataFilter</td></tr>
<tr><th colspan="3">Mapping Rule</th><th>Mapping Type</th></tr>
<tr><td colspan="3">Create container on the receiver side if the ReceiverComSpec contains a Data Filter. Create Container on the sender side if the TransmissionModeCondition element contains a reference to this signal.</td><td>full</td></tr>
</table>

<table>
<tr><th>BSW Module</th><th colspan="3">BSW Context</th></tr>
<tr><td>Com</td><td colspan="3">Com/ComConfig/ComSignalGroup/ComGroupSignal/ComFilter</td></tr>
<tr><th colspan="2">BSW Parameter</th><th colspan="2">BSW Type</th></tr>
<tr><td colspan="2">ComFilterAlgorithm</td><td colspan="2">EcucEnumerationParamDef</td></tr>
<tr><th colspan="4">BSW Description</th></tr>
<tr><td colspan="4">The range of values is specified in the [17] specification, chapter 2.2.2, Reception Filtering.</td></tr>
<tr><th>M2 Template</th><th colspan="3">M2 Description</th></tr>
<tr><td>SWComponentTemplate and SystemTemplate</td><td colspan="3">This attriburte specifies the type of the filter.</td></tr>
<tr><th colspan="4">M2 Parameter</th></tr>
<tr><td colspan="4">CommonStructure::DataFilter.dataFilterType</td></tr>
<tr><th colspan="3">Mapping Rule</th><th>Mapping Type</th></tr>
<tr><td colspan="3">Mapping between DataFilterTypeEnum and ComFilterAlgorithm Enum is necessary.</td><td>full</td></tr>
</table>

<table>
<tr><th>BSW Module</th><th colspan="3">BSW Context</th></tr>
<tr><td>Com</td><td colspan="3">Com/ComConfig/ComSignalGroup/ComGroupSignal/ComFilter</td></tr>
<tr><th colspan="2">BSW Parameter</th><th colspan="2">BSW Type</th></tr>
<tr><td colspan="2">ComFilterMask</td><td colspan="2">EcucIntegerParamDef</td></tr>
<tr><th colspan="4">BSW Description</th></tr>
<tr><td colspan="4">The name of this attribute corresponds to the parameter name in the [17] specification of Reception Filtering.</td></tr>
<tr><th>M2 Template</th><th colspan="3">M2 Description</th></tr>
<tr><td>SWComponentTemplate and SystemTemplate</td><td colspan="3">mask for old and new value</td></tr>
<tr><th colspan="4">M2 Parameter</th></tr>
<tr><td colspan="4">CommonStructure::DataFilter.mask</td></tr>
<tr><th colspan="3">Mapping Rule</th><th>Mapping Type</th></tr>
<tr><td colspan="3">1:1 mapping</td><td>full</td></tr>
</table>

— AUTOSAR CONFIDENTIAL —

BSW Module	BSW Context	
Com	Com/ComConfig/ComSignalGroup/ComGroupSignal/ComFilter	
BSW Parameter		**BSW Type**
ComFilterMax		EcucIntegerParamDef
BSW Description		
The name of this attribute corresponds to the parameter name in the [17] specification of Reception Filtering.		
M2 Template	**M2 Description**	
SWComponentTemplate and SystemTemplate	Value to specify the upper boundary	
M2 Parameter		
CommonStructure::DataFilter::max		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
Com	Com/ComConfig/ComSignalGroup/ComGroupSignal/ComFilter	
BSW Parameter		**BSW Type**
ComFilterMin		EcucIntegerParamDef
BSW Description		
The name of this attribute corresponds to the parameter name in the [17] specification of Reception Filtering.		
M2 Template	**M2 Description**	
SystemTemplate and SWComponentTemplate	Value to specify the lower boundary	
M2 Parameter		
CommonStruture::DataFilter.min		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context	
Com	Com/ComConfig/ComSignalGroup/ComGroupSignal/ComFilter	
BSW Parameter		**BSW Type**
ComFilterOffset		EcucIntegerParamDef
BSW Description		
The name of this attribute corresponds to the parameter name in the [17] specification of Reception Filtering. Range = 0..(ComFilterPeriod-1)		
M2 Template	**M2 Description**	
SWComponentTemplate and SystemTemplate	Specifies the initial number of messages to occur before the first message is passed	
M2 Parameter		
CommonStructure::DataFilter.offset		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context

Com	Com/ComConfig/ComSignalGroup/ComGroupSignal/ComFilter		
BSW Parameter		**BSW Type**	
ComFilterPeriod		EcucIntegerParamDef	
BSW Description			
This parameter defines the period of the ComFilterAlgorithm ONE_EVERY_N.			
M2 Template	**M2 Description**		
SWComponentTemplate and SystemTemplate	specifies number of messages to occur before the message is passed again		
M2 Parameter			
CommonStructure::DataFilter::period			
Mapping Rule			**Mapping Type**
1:1 mapping			full

BSW Module	**BSW Context**		
Com	Com/ComConfig/ComSignalGroup/ComGroupSignal/ComFilter		
BSW Parameter		**BSW Type**	
ComFilterX		EcucIntegerParamDef	
BSW Description			
The name of this attribute corresponds to the parameter name in the [17] specification of Reception Filtering.			
M2 Template	**M2 Description**		
SystemTemplate and SWComponentTemplate	Value to compare with		
M2 Parameter			
CommonStructure::DataFilter.x			
Mapping Rule			**Mapping Type**
1:1 mapping			full

BSW Module	**BSW Context**		
Com	Com/ComConfig/ComSignalGroup/ComGroupSignal		
BSW Parameter		**BSW Type**	
ComHandleId		EcucIntegerParamDef	
BSW Description			
The numerical value used as the ID. For signals it is required by the API calls Com_UpdateShadowSignal, Com_ReceiveShadowSignal and Com_InvalidateShadowSignal. For signals groups it is required by the Com_SendSignalGroup and Com_ReceiveSignalGroup calls.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	**BSW Context**		
Com	Com/ComConfig/ComSignalGroup/ComGroupSignal		
BSW Parameter		**BSW Type**	

ComSignalDataInvalidValue	EcucStringParamDef
BSW Description	
Defines the data invalid value of the signal. In case the ComSignalType is UINT8, UINT16, UINT32, SINT8, SINT16, SINT32 the string shall be interpreted as defined in the chapter Integer Type in the AUTOSAR EcuC specification. In case the ComSignalType is FLOAT32, FLOAT64 the string shall be interpreted as defined in the chapter Float Type in the AUTOSAR EcuC specification. In case the ComSignalType is BOOLEAN the string shall be interpreted as defined in the chapter Boolean Type in the AUTOSAR EcuC specification. In case the ComSignal is a UINT8_N, UINT6_DYN the string shall be interpreted as a decimal representation of the characters separated by blanks, e.g. "97 98 100" means a string "abd", where the char "a" is in byte 0(lowest address), "b" is in byte 1, and "d" is in byte 2 and (highest address).	
M2 Template	**M2 Description**
SystemTemplate	Optional value to express invalidity of the actual data element.
M2 Parameter	
SystemTemplate::CoreCommunication::ISignal::swDataDefProps.invalidValue	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	**BSW Context**
Com	Com/ComConfig/ComSignalGroup/ComGroupSignal
BSW Parameter	**BSW Type**
ComSignalEndianness	EcucEnumerationParamDef
BSW Description	
Defines the endianness of the signal's network representation.	
M2 Template	**M2 Description**
SystemTemplate	This parameter defines the order of the bytes of the signal and the packing into the IPdu. The byte ordering Little Endian (MostSignificantByteLast), Big Endian (MostSignificantByteFirst) and Opaque can be selected.
M2 Parameter	
Fibex::FibexCore::CoreCommunication::ISignalToIPduMapping.packingByteOrder	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	**BSW Context**
Com	Com/ComConfig/ComSignalGroup/ComGroupSignal
BSW Parameter	**BSW Type**
ComSignalInitValue	EcucStringParamDef
BSW Description	
Initial value for this signal. The default value is 0. In case the ComSignalType is UINT8, UINT16, UINT32, SINT8, SINT16, SINT32 the string shall be interpreted as defined in the chapter Integer Type in the AUTOSAR EcuC specification. In case the ComSignalType is FLOAT32, FLOAT64 the string shall be interpreted as defined in the chapter Float Type in the AUTOSAR EcuC specification. In case the ComSignalType is BOOLEAN the string shall be interpreted as defined in the chapter Boolean Type in the AUTOSAR EcuC specification. In case the ComSignal is a UINT8_N, UINT6_DYN the string shall be interpreted as a decimal representation of the characters separated by blanks, e.g. "97 98 100" means a string "abd", where the char "a" is in byte 0(lowest address), "b" is in byte 1, and "d" is in byte 2 and (highest address).	
M2 Template	**M2 Description**

SystemTemplate and SWComponentTemplate	If a full DataMapping exist this information may be available from a configured SenderComSpec and ReceiverComSpec. In case the System Description doesn't use a complete Software ComponentDescription an optional reference from SystemSignal is used.	
M2 Parameter		
SWComponentTemplate::Communcation::ComSpec.initValue or SystemTemplate::Fibex::FibexCore::CoreCommuncation::ISignal.initValue;		
Mapping Rule		**Mapping Type**
It is possible to aggregate an initValue at the level of a ComSpec in the SWC Template. in case the System Description doesn't use a complete Software Component Description (VFB View) the initValue is defined in the System Template.		full

BSW Module	**BSW Context**	
Com	Com/ComConfig/ComSignalGroup/ComGroupSignal	
BSW Parameter		**BSW Type**
ComSignalLength		EcucIntegerParamDef
BSW Description		
Description: For ComSignalType UINT8_N this parameter specifies the length n in bytes. For ComSignalType UINT8_DYN it specifies the maximum length in bytes. For all other types this parameter shall be ignored. Range: 0..8 for normal CAN/ LIN I-PDUs, 0..254 for normal FlexRay I-PDUs, and 0..4095 for I-PDUs with ComIPduType TP.		
M2 Template	**M2 Description**	
SWComponentTemplate	The number of bits that are used to make up the opaque type.	
M2 Parameter		
Datatype::DataTypes::OpaqueType.numberOfBits		
Mapping Rule		**Mapping Type**
Opaque data shall always be of uint8[n] and shall always be mapped to an n-bytes sized signal.		full

BSW Module	**BSW Context**	
Com	Com/ComConfig/ComSignalGroup/ComGroupSignal	
BSW Parameter		**BSW Type**
ComSignalType		EcucEnumerationParamDef
BSW Description		
The AUTOSAR type of the signal. Whether or not the signal is signed or unsigned can be found by examining the value of this attribute. This type could also be used to reserved appropriate storage in AUTOSAR COM.		
M2 Template	**M2 Description**	
SystemTemplate	Describes a reusable data type on the implementation level. This will typically correspond to a typedef in C-code.	
M2 Parameter		
SystemTemplate::Fibex::FibexCore::CoreCommunication::SystemSignal.networkRepresentationProps.swBaseType		
Mapping Rule		**Mapping Type**
Mapping of AUTOSAR data types (defined in the software component description) to COM Signal Types. Mapping rules are described in System Template Specification.		full

BSW Module	BSW Context	
Com	Com/ComConfig/ComSignalGroup/ComGroupSignal	
BSW Parameter		**BSW Type**
ComSystemTemplateSystemSignalRef		EcucForeignReferenceDef
BSW Description		
Reference to the ISignalToIPduMapping that contains a reference to the ISignal (System Template) which this ComSignal (or ComGroupSignal) represents.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**

BSW Module	BSW Context	
Com	Com/ComConfig/ComSignalGroup/ComGroupSignal	
BSW Parameter		**BSW Type**
ComTransferProperty		EcucEnumerationParamDef
BSW Description		
Optionally defines whether this group signal shall contribute to the TRIGGERED_ON_CHANGE transfer property of the signal group. If at least one group signal of a signal group has the "ComTransferProperty" configured all other group signals of that signal group shall have the attribute configured as well.		
M2 Template	**M2 Description**	
SystemTemplate	The triggered transfer property causes immediate transmission of the IPdu, except if transmission mode Periodic or transmission mode NONE is defined for the IPdu. The Pending transfer property does not cause transmission of an I-PDU.	
M2 Parameter		
Fibex::FibexCore::CoreCommunication::ISignalToIPduMapping.transferProperty		
Mapping Rule		**Mapping Type**
ISignalToIPduMapping element contains a reference to the ISignalGroup and contains the attribute "transferProperty"		full

BSW Module	BSW Context	
Com	Com/ComConfig/ComSignalGroup	
BSW Parameter		**BSW Type**
ComHandleId		EcucIntegerParamDef
BSW Description		
The numerical value used as the ID. For signals it is required by the API calls Com_UpdateShadowSignal, Com_ReceiveShadowSignal and Com_InvalidateShadowSignal. For signals groups it is required by the Com_SendSignalGroup and Com_ReceiveSignalGroup calls.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context
Com	Com/ComConfig/ComSignalGroup
BSW Parameter	**BSW Type**
ComInvalidNotification	EcucFunctionNameDef
BSW Description	
Only valid on receiver side: Name of Com_CbkInv callback function to be called. Name of the function which notifies the RTE about the reception of an invalidated signal/ signal group. Only applicable if ComDataInvalidAction is configured to NOTIFY.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	BSW Context
Com	Com/ComConfig/ComSignalGroup
BSW Parameter	**BSW Type**
ComNotification	EcucFunctionNameDef
BSW Description	
On sender side: Name of Com_CbkTxAck callback function to be called. On receiver side: Name of Com_CbkRxAck callback function to be called. If this parameter is omitted no notification shall take place.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	BSW Context
Com	Com/ComConfig/ComSignalGroup
BSW Parameter	**BSW Type**
ComRxDataTimeoutAction	EcucEnumerationParamDef
BSW Description	
This parameter defines the action performed upon expiration of the reception deadline monitoring timer.	
M2 Template	**M2 Description**
SWComponentTemplate	Strategies of handling a reception timeout violation.
M2 Parameter	
SWCTemplate::Communication:ReceiverComSpec::NonqueuedReceiverComSpec.handleTimeout Type	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	BSW Context
Com	Com/ComConfig/ComSignalGroup
BSW Parameter	**BSW Type**
ComSystemTemplateSignalGroupRef	EcucForeignReferenceDef

BSW Description	
Reference to the ISignalToIPduMapping that contains a reference to the ISignalGroup (SystemTemplate) which this ComSignalGroup represents.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**

BSW Module	BSW Context
Com	Com/ComConfig/ComSignalGroup
BSW Parameter	**BSW Type**
ComTimeout	EcucFloatParamDef
BSW Description	
Defines the length of the deadline monitoring timeout period in seconds. The period for the first timeout period can be configured separately by COM183_Conf.	
M2 Template	**M2 Description**
SWComponentTemplate or SystemTemplate	Timeout value in seconds for the reception of the ISignal.
M2 Parameter	
SWCTemplate::Communication::ReceiverComSpec.aliveTimeout and SystemTemplate::Fibex::FibexCore::CoreCommunication::SignalPort.timeout	
Mapping Rule	**Mapping Type**
If a full DataMapping exist for the SystemSignal this information may be available from a configured ReceiverComSpec, in this case the timeout value in ReceiverComSpec override the optional timeout specification in the System Template.	full

BSW Module	BSW Context
Com	Com/ComConfig/ComSignalGroup
BSW Parameter	**BSW Type**
ComTimeoutNotification	EcucFunctionNameDef
BSW Description	
On sender side: Name of Com_CbkTxTOut callback function to be called. On receiver side: Name of Com_CbkRxTOut callback function to be called.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	BSW Context
Com	Com/ComConfig/ComSignalGroup
BSW Parameter	**BSW Type**
ComTransferProperty	EcucEnumerationParamDef
BSW Description	
Defines if a write access to this signal can trigger the transmission of the corresponding I-PDU. If the I-PDU is triggered, depends also on the transmission mode of the corresponding I-PDU.	
M2 Template	**M2 Description**

SystemTemplate	The triggered transfer property causes immediate transmission of the IPdu, except if transmission mode Periodic or transmission mode NONE is defined for the IPdu. The Pending transfer property does not cause transmission of an I-PDU.
M2 Parameter	
Fibex::FibexCore::CoreCommunication::ISignalToIPduMapping.transferProperty	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	**BSW Context**
Com	Com/ComConfig/ComSignalGroup
BSW Parameter	**BSW Type**
ComUpdateBitPosition	EcucIntegerParamDef
BSW Description	
Bit position of update-bit inside I-PDU. If this attribute is omitted then there is no update-bit. This setting must be consistently on sender and on receiver side. Range: 0..63 for CAN and LIN 0..2031 for FlexRay	
M2 Template	**M2 Description**
SystemTemplate	The UpdateIndicationBit indicates to the receivers that the signal (or the signal group) was updated by the sender. Length is always one bit. The UpdateIndicationBitPosition attribute describes the position of the update bit within the SignalIPdu.
M2 Parameter	
Fibex::FibexCore::CoreCommunication::ISignalToIPduMapping.updateIndicationBitPosition	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	**BSW Context**
Com	Com/ComConfig
BSW Parameter	**BSW Type**
ComTimeBase	EcucParamConfContainerDef
BSW Description	
Contains the timebase parameters for Tx, Rx and routing.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**
Com	Com/ComConfig/ComTimeBase
BSW Parameter	**BSW Type**
ComGwTimeBase	EcucFloatParamDef
BSW Description	

The period between successive calls to Com_MainFunctionRouteSignals in seconds. This parameter may be used by the COM generator to transform the values of the signal gateway related timing configuration parameters of the COM module to internal implementation specific counter or tick values. The COM module's internal timing handling is implementation specific. The COM module (generator) might rely on the fact that Com_MainFunctionRouteSignals is scheduled according to the value configured here.	
M2 Template	**M2 Description**
SystemTemplate	The period between successive calls to Com_MainFunctionRouteSignals of the AUTOSAR COM module in seconds.
M2 Parameter	
Fibex::FibexCore::CoreTopology::ECUInstance::COMConfigurationGwTimeBase	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	**BSW Context**
Com	Com/ComConfig/ComTimeBase
BSW Parameter	**BSW Type**
ComRxTimeBase	EcucFloatParamDef
BSW Description	
The period between successive calls to Com_MainFunctionRx in seconds. This parameter may be used by the COM generator to transform the values of the reception related timing configuration parameters of the COM module to internal implementation specific counter or tick values. The COM module's internal timing handling is implementation specific. The COM module (generator) may rely on the fact that Com_MainFunctionRx is scheduled according to the value configured here.	
M2 Template	**M2 Description**
SystemTemplate	The period between successive calls to Com_MainFunctionRx of the AUTOSAR COM module in seconds.
M2 Parameter	
Fibex::FibexCore::CoreTopology::ECUInstance::COMConfigurationRxTimeBase	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	**BSW Context**
Com	Com/ComConfig/ComTimeBase
BSW Parameter	**BSW Type**
ComTxTimeBase	EcucFloatParamDef
BSW Description	
The period between successive calls to Com_MainFunctionTx in seconds. This parameter may be used by the COM generator to transform the values of the transmission related timing configuration parameters of the COM module to internal implementation specific counter or tick values. The COM module's internal timing handling is implementation specific. The COM module (generator) may rely on the fact that Com_MainFunctionTx is scheduled according to the value configured here.	
M2 Template	**M2 Description**
SystemTemplate	The period between successive calls to Com_MainFunctionTx of the AUTOSAR COM module in seconds.
M2 Parameter	
Fibex::FibexCore::CoreTopology::ECUInstance::COMConfigurationTxTimeBase	

— AUTOSAR CONFIDENTIAL —

Mapping Rule	Mapping Type
1:1 mapping	full

BSW Module	BSW Context	
Com	Com	
BSW Parameter		**BSW Type**
ComGeneral		EcucParamConfContainerDef
BSW Description		
Contains the general configuration parameters of the module.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**

BSW Module	BSW Context	
Com	Com/ComGeneral	
BSW Parameter		**BSW Type**
ComConfigurationUseDet		EcucBooleanParamDef
BSW Description		
The error hook shall contain code to call the Det. If this parameter is configured COM_DEV_ERROR_DETECT shall be set to ON as output of the configuration tool. (as input for the source code), see COM028.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
Com	Com/ComGeneral	
BSW Parameter		**BSW Type**
ComEnableMDTForCyclicTransmission		EcucBooleanParamDef
BSW Description		
Enables globally for the whole Com module the minimum delay time monitoring for cyclic and repeated transmissions (ComTxModeMode=PERIODIC or ComTxModeMode=MIXED for the cyclic transmissions, ComTxModeNumberOfRepetitions > 0 for repeated transmissions).		
M2 Template	**M2 Description**	
SystemTemplate	Enables for the Com module of this EcuInstance the minimum delay time monitoring for cyclic and repeated transmissions (TransmissionModeTiming has cyclicTiming assigned or eventControlledTiming with numberOfRepetitions > 0)	
M2 Parameter		
CoreTopology::EcuInstance.comEnableMDTForCyclicTransmission		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context

Com	Com/ComGeneral		
BSW Parameter		**BSW Type**	
ComRetryFailedTransmitRequests		EcucBooleanParamDef	
BSW Description			
If this Parameter is set to true, retry of failed transmission requests is enabled. If this Parameter is not present, the default value is assumed.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	**BSW Context**		
Com	Com/ComGeneral		
BSW Parameter		**BSW Type**	
ComSupportedIPduGroups		EcucIntegerParamDef	
BSW Description			
Defines the maximum number of supported I-PDU groups.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	**BSW Context**		
Com	Com/ComGeneral		
BSW Parameter		**BSW Type**	
ComVersionInfoApi		EcucBooleanParamDef	
BSW Description			
Activate/Deactivate the version information API (Com_GetVersionInfo). True: version information API activated False: version information API deactivated			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

D.18 ComM Mapping

BSW Module	**BSW Context**	
ComM	ComM	
BSW Parameter		**BSW Type**
ComMConfigSet		EcucParamConfContainerDef
BSW Description		
This container is the base for a multiple configuration set.		

M2 Template	M2 Description	
M2 Parameter		
Mapping Rule		**Mapping Type**

BSW Module	BSW Context		
ComM	ComM/ComMConfigSet		
BSW Parameter		**BSW Type**	
ComMChannel		EcucParamConfContainerDef	
BSW Description			
This container contains the configuration (parameters) of the bus channel(s). The channel parameters shall be harmonized within the whole communication stack.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**

BSW Module	BSW Context		
ComM	ComM/ComMConfigSet/ComMChannel		
BSW Parameter		**BSW Type**	
ComMBusType		EcucEnumerationParamDef	
BSW Description			
Identifies the bus type of the channel.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**

BSW Module	BSW Context		
ComM	ComM/ComMConfigSet/ComMChannel		
BSW Parameter		**BSW Type**	
ComMChannelId		EcucIntegerParamDef	
BSW Description			
Channel identification number of the corresponding channel.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**

BSW Module	BSW Context
ComM	ComM/ComMConfigSet/ComMChannel

BSW Parameter		BSW Type
ComMFullCommRequestNotificationEnabled		EcucBooleanParamDef
BSW Description		
Defines if the optional SenderReceiver Port of Interface ComM_CurrentChannelRequest will be provided for this channel. True means enabled. False means disabled		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**

BSW Module	BSW Context		
ComM	ComM/ComMConfigSet/ComMChannel		
BSW Parameter		**BSW Type**	
ComMGlobalNvmBlockDescriptor		EcucBooleanParamDef	
BSW Description			
If this parameter is set to "true", the NoWakeUp inhibition state of the channel shall be stored (in some implementation specific way) in the block pointed to by ComMGlobalNvmBlockDescriptor.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**

BSW Module	BSW Context		
ComM	ComM/ComMConfigSet/ComMChannel		
BSW Parameter		**BSW Type**	
ComMMainFunctionPeriod		EcucFloatParamDef	
BSW Description			
Specifies the period in seconds that the MainFunction has to be triggered with. Comment: ComM scheduling shall be at least as fast as the communication stack and a schedule longer than 100ms makes no sense for communication.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**

BSW Module	BSW Context	
ComM	ComM/ComMConfigSet/ComMChannel	
BSW Parameter		**BSW Type**
ComMNetworkManagement		EcucParamConfContainerDef
BSW Description		
This container contains the configuration parameters of the networkmanagement.		
M2 Template	**M2 Description**	

M2 Parameter	
Mapping Rule	**Mapping Type**

BSW Module	**BSW Context**	
ComM	ComM/ComMConfigSet/ComMChannel/ComMNetworkManagement	
BSW Parameter		**BSW Type**
ComMNmLightTimeout		EcucFloatParamDef
BSW Description		
Defines the timeout (in seconds) after COMM_FULL_COMMUNICATION sub-state COMM_FULL_COM_READY_SLEEP is left. The range shall be greater than 0.0 and less or equal to 255.0.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**

BSW Module	**BSW Context**	
ComM	ComM/ComMConfigSet/ComMChannel/ComMNetworkManagement	
BSW Parameter		**BSW Type**
ComMNmVariant		EcucEnumerationParamDef
BSW Description		
Defines the functionality of the networkmanagement. Shall be harmonized with NM configuration.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**

BSW Module	**BSW Context**	
ComM	ComM/ComMConfigSet/ComMChannel/ComMNetworkManagement	
BSW Parameter		**BSW Type**
ComMPncNmRequest		EcucBooleanParamDef
BSW Description		
If this parameter equals true then every time a FULL Communication is requested due to a change in the PNC state machine to PNC_REQUESTED Nm shall be called using the API Nm_NetworkRequest.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**

BSW Module	BSW Context	
ComM	ComM/ComMConfigSet/ComMChannel	
BSW Parameter		**BSW Type**
ComMNoCom		EcucBooleanParamDef
BSW Description		
Not allowed to change state of ComM channel to COMM_SILENT_COMMUNICATION or COMM_FULL_COMMUNICATION. true: Enabled - Not allowed to switch to Communication Modes above. false: Disabled - Allowed to switch Communication Modes above. Shall be possible to change parameter during runtime with ComM API's. ECU/All channels: ComM_LimitECUToNoComMode(). Separate channels: ComM_LimitChannelToNoComMode().		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**

BSW Module	BSW Context	
ComM	ComM/ComMConfigSet/ComMChannel	
BSW Parameter		**BSW Type**
ComMNoWakeup		EcucBooleanParamDef
BSW Description		
Defines if an ECU is not allowed to wake-up the channel. true: Enabled (not allowed to wake-up)) false: Disabled This is the default/init value of a runtime variable that can be changed during runtime using ComM_PreventWakeUp().		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**

BSW Module	BSW Context	
ComM	ComM/ComMConfigSet/ComMChannel	
BSW Parameter		**BSW Type**
ComMPncGatewayType		EcucEnumerationParamDef
BSW Description		
Identifies the Partial Network Gateway behaviour of a ComMChannel.		
M2 Template	**M2 Description**	
SystemTemplate		
M2 Parameter		
CoreTopology::CommunicationConnector.pncGatewayType		
Mapping Rule		**Mapping Type**
1:1 mapping		full

— AUTOSAR CONFIDENTIAL —

BSW Module	BSW Context	
ComM	ComM/ComMConfigSet/ComMChannel	
BSW Parameter		**BSW Type**
ComMUserPerChannel		EcucParamConfContainerDef
BSW Description		
This container contains a list of identifiers that are needed to refer to a user in the system which is linked to a channel.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**

BSW Module	BSW Context	
ComM	ComM/ComMConfigSet/ComMChannel/ComMUserPerChannel	
BSW Parameter		**BSW Type**
ComMUserChannel		EcucReferenceDef
BSW Description		
Reference to the ComMUser that corresponds to this channel user. ImplementationType: COMM_UserHandleType		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**

BSW Module	BSW Context	
ComM	ComM/ComMConfigSet	
BSW Parameter		**BSW Type**
ComMPnc		EcucParamConfContainerDef
BSW Description		
This container contains the configuration of the partial network cluster (PNC).		
M2 Template	**M2 Description**	
SystemTemplate	Describes a mapping between one or several Virtual Function Clusters onto Partial Network Clusters. A Virtual Function Cluster is realized by a PortGroup. A Partial Network Cluster is realized by one or more IPduGroups.	
M2 Parameter		
SWmapping::PncMapping		
Mapping Rule		**Mapping Type**
Create ComMPnc container for each PncMapping element.		full

BSW Module	BSW Context	
ComM	ComM/ComMConfigSet/ComMPnc	
BSW Parameter		**BSW Type**
ComMChannelPerPnc		EcucReferenceDef
BSW Description		

Reference to the ComMChannel that is required for this PNC. ImplementationType: COMM_ChannelType		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**

BSW Module	**BSW Context**	
ComM	ComM/ComMConfigSet/ComMPnc	
BSW Parameter		**BSW Type**
ComMPncComSignal		EcucParamConfContainerDef
BSW Description		
Represents the PncComSignals which are used to communicate the EIRA and ERA status of this PNC.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**

BSW Module	**BSW Context**	
ComM	ComM/ComMConfigSet/ComMPnc/ComMPncComSignal	
BSW Parameter		**BSW Type**
ComMPncComSignalChannelRef		EcucReferenceDef
BSW Description		
Reference to the ComMChannel which is used to determine whether this PncComSignal shall participate in the active or passive role (via the parameter ComMPncGatewayType of the ComMChannel). Not applicable if ComMPncComSignalKind is EIRA.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**

BSW Module	**BSW Context**	
ComM	ComM/ComMConfigSet/ComMPnc/ComMPncComSignal	
BSW Parameter		**BSW Type**
ComMPncComSignalDirection		EcucEnumerationParamDef
BSW Description		
Indicates the communication direction of this PncComSignal.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**

BSW Module	BSW Context
ComM	ComM/ComMConfigSet/ComMPnc/ComMPncComSignal
BSW Parameter	**BSW Type**
ComMPncComSignalKind	EcucEnumerationParamDef
BSW Description	
Indicates whether this PncComSignal represents EIRA or ERA PNC information.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**

BSW Module	BSW Context
ComM	ComM/ComMConfigSet/ComMPnc/ComMPncComSignal
BSW Parameter	**BSW Type**
ComMPncComSignalRef	EcucSymbolicNameReferenceDef
BSW Description	
Reference to the ComSignal which is used to transport the partial network channel request information.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**

BSW Module	BSW Context
ComM	ComM/ComMConfigSet/ComMPnc
BSW Parameter	**BSW Type**
ComMPncId	EcucIntegerParamDef
BSW Description	
Partial network cluster identification number.	
M2 Template	**M2 Description**
SystemTemplate	Identifer of the Partial Network Cluster.
M2 Parameter	
SWmapping::PncMapping.pncIdentifier	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	BSW Context
ComM	ComM/ComMConfigSet/ComMPnc
BSW Parameter	**BSW Type**
ComMUserPerPnc	EcucReferenceDef
BSW Description	

Reference to the ComMUsers that correspond to this PNC. ImplementationType: COMM_UserHandleType	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**

BSW Module	**BSW Context**	
ComM	ComM/ComMConfigSet	
BSW Parameter		**BSW Type**
ComMPncEnabled		EcucBooleanParamDef
BSW Description		
Defines whether in this configuration set the partial networking is enabled. true: Enabled false: Disabled		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**

BSW Module	**BSW Context**	
ComM	ComM/ComMConfigSet	
BSW Parameter		**BSW Type**
ComMUser		EcucParamConfContainerDef
BSW Description		
This container contains a list of identifiers that are needed to refer to a user in the system which is designated to request Communication modes.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**

BSW Module	**BSW Context**	
ComM	ComM/ComMConfigSet/ComMUser	
BSW Parameter		**BSW Type**
ComMUserEcucPartitionRef		EcucReferenceDef
BSW Description		
Denotes in which "EcucPartition" the requester is executed. When the partition is stopped, the communication request shall be cancelled in the ComM to avoid a stay-awake situation of the bus due to a stopped partition.		
M2 Template	**M2 Description**	

M2 Parameter	
Mapping Rule	**Mapping Type**

BSW Module	**BSW Context**	
ComM	ComM/ComMConfigSet/ComMUser	
BSW Parameter		**BSW Type**
ComMUserIdentifier		EcucIntegerParamDef
BSW Description		
An identifier that is needed to refer to a user in the system which is designated to request Communication Modes. ImplementationType: ComM_UserHandleType		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**

BSW Module	**BSW Context**	
ComM	ComM	
BSW Parameter		**BSW Type**
ComMGeneral		EcucParamConfContainerDef
BSW Description		
General configuration parameters of the Communication Manager.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**

BSW Module	**BSW Context**	
ComM	ComM/ComMGeneral	
BSW Parameter		**BSW Type**
ComMDevErrorDetect		EcucBooleanParamDef
BSW Description		
Switches the Development Error Detection and Notification ON or OFF. true: Enabled false: Disabled		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**

BSW Module	BSW Context	
ComM	ComM/ComMGeneral	
BSW Parameter		**BSW Type**
ComMDirectUserMapping		EcucBooleanParamDef
BSW Description		
If this parameter is set to true the configuration tool shall automatically create a ComMUser per ComMPnc and a ComMUser per ComMChannel. The shortName of the generated ComMUsers shall follow the following naming convention: PNCUser_ComMPncId, e.g. PNCUser_13 ChannelUser_ComMChannelId, e.g. ChannelUser_25 Restriction: ComMUser, which are created due to this configuration parameter, shall not be used by SWCs (only available for BswM).		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**

BSW Module	BSW Context	
ComM	ComM/ComMGeneral	
BSW Parameter		**BSW Type**
ComMEcuGroupClassification		EcucIntegerParamDef
BSW Description		
Defines whether a mode inhibition affects the ECU or not. Examples: 000: No mode inhibition can be activated 001: Wake up inhibition can be enabled Forcing into COMM_NO_COMMUNICATION mode shall be switched on if ComMNmVariant=PASSIVE.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**

BSW Module	BSW Context	
ComM	ComM/ComMGeneral	
BSW Parameter		**BSW Type**
ComMGlobalNvMBlockDescriptor		EcucSymbolicNameReferenceDef
BSW Description		
Reference to NVRAM block containing the none volatile data. If this parameter is not configured it means that no NVRam is used at all.		
M2 Template	**M2 Description**	

M2 Parameter	
Mapping Rule	**Mapping Type**

BSW Module	**BSW Context**	
ComM	ComM/ComMGeneral	
BSW Parameter		**BSW Type**
ComMModeLimitationEnabled		EcucBooleanParamDef
BSW Description		
true if mode limitation functionality shall be enabled. true: Enabled false: Disabled		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**

BSW Module	**BSW Context**	
ComM	ComM/ComMGeneral	
BSW Parameter		**BSW Type**
ComMPncGatewayEnabled		EcucBooleanParamDef
BSW Description		
Enables or disables support of Partial Network Gateway. False: Partial Networking Gateway is disabled True: Partial Networking Gateway is enabled		
M2 Template	**M2 Description**	
SystemTemplate		
M2 Parameter		
CoreTopology::CommunicationConnector.pncGatewayType		
Mapping Rule		**Mapping Type**
Enabed if at least one CommunicationConnector of this EcuInstance has the pncGatewayType set to active or passive.		full

BSW Module	**BSW Context**	
ComM	ComM/ComMGeneral	
BSW Parameter		**BSW Type**
ComMPncPrepareSleepTimer		EcucFloatParamDef
BSW Description		
Time in seconds the PNC state machine shall wait in PNC_PREPARE_SLEEP.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**

BSW Module	BSW Context		
ComM	ComM/ComMGeneral		
BSW Parameter		**BSW Type**	
ComMPncSupport		EcucBooleanParamDef	
BSW Description			
Enables or disables support of partial networking. False: Partial Networking is disabled True: Partial Networking is enabled			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**

BSW Module	BSW Context		
ComM	ComM/ComMGeneral		
BSW Parameter		**BSW Type**	
ComMSynchronousWakeUp		EcucBooleanParamDef	
BSW Description			
Wake up of one channel shall lead to a wake up of all channels if true. true: Enabled false: Disabled			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**

BSW Module	BSW Context		
ComM	ComM/ComMGeneral		
BSW Parameter		**BSW Type**	
ComMTMinFullComModeDuration		EcucFloatParamDef	
BSW Description			
Minimum time duration in seconds, spent in the COMM_FULL_COMMUNICATION sub-state COMM_FULL_COM_NETWORK_REQUESTED.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**

BSW Module	BSW Context		
ComM	ComM/ComMGeneral		
BSW Parameter		**BSW Type**	
ComMVersionInfoApi		EcucBooleanParamDef	

— AUTOSAR CONFIDENTIAL —

BSW Description		
Switches the possibility to read the published information with the service ComM_GetPublishedInformation(). true: Enabled false: Disabled		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**

BSW Module	BSW Context	
ComM	ComM/ComMGeneral	
BSW Parameter		**BSW Type**
ComMWakeupInhibitionEnabled		EcucBooleanParamDef
BSW Description		
true if wake up inhibition functionality enabled. true: Enabled false: Disabled		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**

D.19 PduR Mapping

BSW Module	BSW Context	
PduR	PduR	
BSW Parameter		**BSW Type**
PduRBswModules		EcucParamConfContainerDef
BSW Description		
Each container describes a specific BSW module (upper/CDD/lower/IpduM) that the PDU Router shall interface to. The reason to have it as own configuration container instead of implication of the routing path is to be able to configure CDD:s properly and to force module's to be used in a post-build situation even though no routing is made to/from this module (future configurations may include these modules).		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
PduR	PduR/PduRBswModules	
BSW Parameter		**BSW Type**
PduRBswModuleRef		EcucForeignReferenceDef
BSW Description		
This is a reference to one BSW module's configuration (i.e. not the ECUC parameter definition template). Example, there could be several configurations of LinIf and this reference selects one of them.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
PduR	PduR/PduRBswModules	
BSW Parameter		**BSW Type**
PduRCancelReceive		EcucBooleanParamDef
BSW Description		
Specifies if the Transport protocol module supports the CancelReceive API or not. Value true the API is supported.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
PduR	PduR/PduRBswModules	
BSW Parameter		**BSW Type**
PduRCancelTransmit		EcucBooleanParamDef
BSW Description		
Specifies if the BSW module supports the CancelTransmit API or not. Value true the API is supported.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
PduR	PduR/PduRBswModules	
BSW Parameter		**BSW Type**
PduRChangeParameterRequestApi		EcucBooleanParamDef
BSW Description		
This parameter, if set to true, enables the PduR_<Up>ChangeParameterRequest Api for this Module.		

M2 Template	M2 Description	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context		
PduR	PduR/PduRBswModules		
BSW Parameter		**BSW Type**	
PduRCommunicationInterface		EcucBooleanParamDef	
BSW Description			
Specifies if the BSW module supports the Communication Interface APIs or not. Value true the APIs are supported. A module can have both Communication Interface APIs and Transport Protocol APIs (e.g. the COM module).			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	BSW Context		
PduR	PduR/PduRBswModules		
BSW Parameter		**BSW Type**	
PduRLowerModule		EcucBooleanParamDef	
BSW Description			
The PduRLowerModule will decide who will call the APIs and who will implement the APIs. For example, if the CanIf module is referenced then the PDU Router module will implement the PduR_CanIfRxIndication API. And the PDUR module will call the CanIf_Transmit API. Other APIs are of course also covered. An upper module can also be an lower module (e.g. the IpduM module).			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	BSW Context	
PduR	PduR/PduRBswModules	
BSW Parameter		**BSW Type**
PduRRetransmission		EcucBooleanParamDef
BSW Description		

— AUTOSAR CONFIDENTIAL —

If set to true this means that the destination transport protocol module will use the retransmission feature. This parameter might be set to false if the retransmission feature is not used, even though the destination transport protocol is supporting it. This parameter is only valid for transport protocol modules and gateway operations. If transmission from a local upper layer module this module will handle the retransmission.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
PduR	PduR/PduRBswModules	
BSW Parameter		**BSW Type**
PduRTransportProtocol		EcucBooleanParamDef
BSW Description		
The PDU Router module shall use the API parameters specified for transport protocol interface.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
PduR	PduR/PduRBswModules	
BSW Parameter		**BSW Type**
PduRTriggertransmit		EcucBooleanParamDef
BSW Description		
Specifies if the BSW module supports the TriggerTransmit API or not. Value true the API is supported.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
PduR	PduR/PduRBswModules	
BSW Parameter		**BSW Type**
PduRTxConfirmation		EcucBooleanParamDef
BSW Description		
Specifies if the BSW module supports the TxConfirmation API or not. Value true the API is supported.		
M2 Template	**M2 Description**	
M2 Parameter		

— AUTOSAR CONFIDENTIAL —

Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**	
PduR	PduR/PduRBswModules	
BSW Parameter		**BSW Type**
PduRUpperModule		EcucBooleanParamDef
BSW Description		
The PduRUpperModule will decide who will call the APIs and who will implement the APIs. For example, if the COM module is referenced then the PDU Router module will implement the PduR_Transmit API. And the PDUR module will call the Com_RxIndication API. Other APIs are of course also covered. An upper module can also be an lower module (e.g. the IpduM module).		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
PduR	PduR/PduRBswModules	
BSW Parameter		**BSW Type**
PduRUseTag		EcucBooleanParamDef
BSW Description		
This parameter, if set to true, enables the usage of the tag (<up>) in the following API calls: * PduR_<Up>CancelReceiveRequest * PduR_<Up>CancelTransmitRequest * PduR_<Up>ChangeParameterRequest Example: If used by COM and the parameter is enabled the PduR_ComCancelTransmitRequest is used. The background is that upper layer modules differ in usage of this tag (e.g. COM is using the tag, DCM is not).		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
PduR	PduR	
BSW Parameter		**BSW Type**
PduRGeneral		EcucParamConfContainerDef
BSW Description		

This container is a subcontainer of PduR and specifies the general configuration parameters of the PDU Router.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**	
PduR	PduR/PduRGeneral	
BSW Parameter		**BSW Type**
PduRDevErrorDetect		EcucBooleanParamDef
BSW Description		
If true then PDU Router will enable the error-reporting to the Development Error Tracer (DET).		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
PduR	PduR/PduRGeneral	
BSW Parameter		**BSW Type**
PduRVersionInfoApi		EcucBooleanParamDef
BSW Description		
If true the PduR_GetVersionInfo API is available.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
PduR	PduR/PduRGeneral	
BSW Parameter		**BSW Type**
PduRZeroCostOperation		EcucBooleanParamDef
BSW Description		
If set the PduR configuration generator will report an error if zero-cost-operation cannot be fulfilled. This parameter shall be seen as an input requirement to the configuration generator.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
PduR	PduR	
BSW Parameter		**BSW Type**
PduRRoutingTables		EcucParamConfContainerDef
BSW Description		
Represents one table of routing paths. This routing table allows multiple configurations that can be used to create several routing tables in the same configuration. This is mainly used for post-build (e.g. post-build selectable) but can be used by pre-compile and link-time for variant handling.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
PduR	PduR/PduRRoutingTables	
BSW Parameter		**BSW Type**
PduRConfigurationId		EcucIntegerParamDef
BSW Description		
Identification of the configuration of the PduR configuration. This identification can be read using the PduR API.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**

BSW Module	BSW Context	
PduR	PduR/PduRRoutingTables	
BSW Parameter		**BSW Type**
PduRRoutingPathGroup		EcucParamConfContainerDef
BSW Description		
This container groups routing path destinations. Destinations are used instead of routing paths since a routing path can be 1:n. It is desirable to be able to enable/disable a specific bus (i.e. a destination) rather than a routing path. Of course it is possible to create groups that covers specific routing paths as well. Enabling and disabling of routing path groups are made using the PduR API		
M2 Template	**M2 Description**	
SystemTemplate	The AUTOSAR PduR will enable and disable the sending of configurable groups of I-Pdus during runtime according to the AUTOSAR PduR specification.	
M2 Parameter		
Fibex::FibexCore::CoreCommunication::PduRIPduGroup		
Mapping Rule		**Mapping Type**
Create container for each existing PduRIPduGroup that is connected to the regarded Ecu		full

BSW Module	BSW Context	
PduR	PduR/PduRRoutingTables/PduRRoutingPathGroup	
BSW Parameter		**BSW Type**
PduRDestPduRef		EcucReferenceDef
BSW Description		
This reference selects one destination of the routing path.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
PduR	PduR/PduRRoutingTables/PduRRoutingPathGroup	
BSW Parameter		**BSW Type**
PduRIsEnabledAtInit		EcucBooleanParamDef
BSW Description		
If set to true this routing path group will be enabled after initializing the PDU Router module (i.e. enabled in the PduR_Init function).		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
PduR	PduR/PduRRoutingTables/PduRRoutingPathGroup	
BSW Parameter		**BSW Type**
PduRRoutingPathGroupId		EcucIntegerParamDef
BSW Description		
Identification of the routing group. The identification will be used by the disable/enable API in the PDU Router module API.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
PduR	PduR/PduRRoutingTables	
BSW Parameter		**BSW Type**
PduRRoutingTable		EcucParamConfContainerDef
BSW Description		
Represents one container of routing paths. Each container is either minimum routing or not.		
M2 Template	**M2 Description**	
SystemTemplate		

M2 Parameter	
Fibex::Fibex4Multiplatform::IPduMapping or Fibex::FibexCore::CoreCommunication::PduTriggering or TransportProtocols::TpConfig	
Mapping Rule	**Mapping Type**
For each MultiplatformGateway.pduMapping; for each SignalPdu-Multiplexed Pdu Connection; for each IPduTriggering; for each TpConfig create one Pdu RRoutingPath.	full

BSW Module	BSW Context	
PduR	PduR/PduRRoutingTables/PduRRoutingTable	
BSW Parameter		**BSW Type**
PduRIsMinimumRouting		EcucBooleanParamDef
BSW Description		
Specifies if the container contains routing paths that are of the type minimum routing or not.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
PduR	PduR/PduRRoutingTables/PduRRoutingTable	
BSW Parameter		**BSW Type**
PduRRoutingPath		EcucParamConfContainerDef
BSW Description		
This container is a subcontainer of PduRRoutingTable and specifies the routing path of a PDU.		
M2 Template	**M2 Description**	
SystemTemplate		
M2 Parameter		
Fibex::Fibex4Multiplatform::IPduMapping or Fibex::FibexCore::CoreCommunication::PduTriggering or TransportProtocols::TpConfig		
Mapping Rule		**Mapping Type**
For each MultiplatformGateway.pduMapping; for each SignalPdu-Multiplexed Pdu Connection; for each IPduTriggering; for each TpConfig create one Pdu RRoutingPath.		full

BSW Module	BSW Context	
PduR	PduR/PduRRoutingTables/PduRRoutingTable/PduRRoutingPath	
BSW Parameter		**BSW Type**
PduRDestPdu		EcucParamConfContainerDef
BSW Description		
This container is a subcontainer of PduRRoutingPath and specifies one destination for the PDU to be routed.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context		
PduR	PduR/PduRRoutingTables/PduRRoutingTable/PduRRoutingPath/PduRDest Pdu		
BSW Parameter		**BSW Type**	
PduRDefaultValue		EcucParamConfContainerDef	
BSW Description			
Specifies the default value of the I-PDU. Only required for gateway operation and if at least one PDU specified by PduRDestPdu uses TriggerTransmit Data provision. Represented as an array of IntegerParamDef.			
M2 Template	**M2 Description**		
SystemTemplate	Default Value which will be distributed if no pdu has been received since last		
M2 Parameter			
Fibex::Fibex4Multiplatform::IPduMapping::PduMappingDefaultValue			
Mapping Rule			**Mapping Type**
Container should be created if PduMappingDefaulValue is described in the Sys-T			full

BSW Module	BSW Context		
PduR	PduR/PduRRoutingTables/PduRRoutingTable/PduRRoutingPath/PduRDest Pdu/PduRDefaultValue		
BSW Parameter		**BSW Type**	
PduRDefaultValueElement		EcucParamConfContainerDef	
BSW Description			
Each value element is represented by the element and the position in an array.			
M2 Template	**M2 Description**		
SystemTemplate	The default value consists of a number of elements. Each element is one byte long and the number of elements is specified by SduLength.		
M2 Parameter			
Fibex::Fibex4Multiplatform::DefaultValueElement			
Mapping Rule			**Mapping Type**
Container must be created for each DefaultValueElement that is aggregated by PduMappingDefaultValue			full

BSW Module	BSW Context		
PduR	PduR/PduRRoutingTables/PduRRoutingTable/PduRRoutingPath/PduRDest Pdu/PduRDefaultValue/PduRDefaultValueElement		
BSW Parameter		**BSW Type**	
PduRDefaultValueElement		EcucIntegerParamDef	
BSW Description			
The default value consists of a number of elements. Each element is one byte long and the number of elements is specified by SduLength. The position of this parameter in the container is specified by the PduRElementBytePosition parameter.			
M2 Template	**M2 Description**		
SystemTemplate	The integer value of a freely defined data byte.		
M2 Parameter			
Fibex::Fibex4Multiplatform::DefaultValueElement.elementByteValue			
Mapping Rule			**Mapping Type**
1:1 mapping			full

BSW Module	BSW Context

PduR	PduR/PduRRoutingTables/PduRRoutingTable/PduRRoutingPath/PduRDestPdu/PduRDefaultValue/PduRDefaultValueElement
BSW Parameter	**BSW Type**
PduRDefaultValueElementBytePosition	EcucIntegerParamDef
BSW Description	
This parameter specifies the byte position of the element within the default value	
M2 Template	**M2 Description**
SystemTemplate	The default value consists of a number of elements. Each element is one byte long and the number of elements is specified by SduLength.
M2 Parameter	
Fibex::Fibex4Multiplatform::DefaultValueElement..elementPosition	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	**BSW Context**
PduR	PduR/PduRRoutingTables/PduRRoutingTable/PduRRoutingPath/PduRDestPdu
BSW Parameter	**BSW Type**
PduRDestPduDataProvision	EcucEnumerationParamDef
BSW Description	
Specifies how data are provided: direct (as part of the Transmit call) or via the TriggerTransmit callback function. Only required for non-TP I-PDUs (local and gatewayed).	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**
PduR	PduR/PduRRoutingTables/PduRRoutingTable/PduRRoutingPath/PduRDestPdu
BSW Parameter	**BSW Type**
PduRDestPduHandleId	EcucIntegerParamDef
BSW Description	
PDU identifier assigned by PDU Router. Used by communication interface and transport protocol modules for confirmation.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**
PduR	PduR/PduRRoutingTables/PduRRoutingTable/PduRRoutingPath/PduRDestPdu
BSW Parameter	**BSW Type**
PduRDestPduRef	EcucReferenceDef
BSW Description	

Destination PDU reference; reference to unique PDU identifier which shall be used by the PDU Router instead of the source PDU ID when calling the related function of the destination module.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**
PduR	PduR/PduRRoutingTables/PduRRoutingTable/PduRRoutingPath/PduRDestPdu
BSW Parameter	**BSW Type**
PduRDestTxBufferRef	EcucReferenceDef
BSW Description	
Reference to a buffer that is allocated in the PduRTxBuffer. Having a global (for PduR) list of buffers allows reusage and hence less memory consumption.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**
PduR	PduR/PduRRoutingTables/PduRRoutingTable/PduRRoutingPath/PduRDestPdu
BSW Parameter	**BSW Type**
PduRTpThreshold	EcucIntegerParamDef
BSW Description	
Defines the number of bytes which shall be received before transmission on the destination bus may start. Only required for routing-on-the-fly TP gateway PDUs. The threshold shall not be larger than the length of the related TP Buffer.	
M2 Template	**M2 Description**
SystemTemplate	Optionally defines the to be configured Pdu Router TpChunkSize for
M2 Parameter	
ibex::Fibex4Multiplatform::IPduMapping.pdurTpChunkSize	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	**BSW Context**
PduR	PduR/PduRRoutingTables/PduRRoutingTable/PduRRoutingPath/PduRDestPdu
BSW Parameter	**BSW Type**
PduRTransmissionConfirmation	EcucBooleanParamDef
BSW Description	

This parameter is only for communication interfaces. Transport protocol modules will always call the TxConfirmation function. If set the destination communication interface module will call the TxConfirmation. However the TxConfirmation may be not called due to error. So the PduR shall not block until the TxConfirmation is called. One background for this parameter is for the PduR to know when all modules have confirmed a multicast operation.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**
PduR	PduR/PduRRoutingTables/PduRRoutingTable/PduRRoutingPath
BSW Parameter	**BSW Type**
PduRSrcPdu	EcucParamConfContainerDef
BSW Description	
This container is a subcontainer of PduRRoutingPath and specifies the source of the PDU to be routed.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**
PduR	PduR/PduRRoutingTables/PduRRoutingTable/PduRRoutingPath/PduRSrcPdu
BSW Parameter	**BSW Type**
PduRSourcePduHandleId	EcucIntegerParamDef
BSW Description	
PDU identifier assigned by PDU Router.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**
PduR	PduR/PduRRoutingTables/PduRRoutingTable/PduRRoutingPath/PduRSrcPdu
BSW Parameter	**BSW Type**
PduRSrcPduRef	EcucReferenceDef
BSW Description	
Source PDU reference; reference to unique PDU identifier which shall be used for the requested PDU Router operation.	
M2 Template	**M2 Description**

M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**	
PduR	PduR/PduRRoutingTables	
BSW Parameter		**BSW Type**
PduRTpBufferTable		EcucParamConfContainerDef
BSW Description		
This container will specify the needed buffers for gatewaying using TP. It is not connected to the specific routing path destination to allow a more efficient buffer handling.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
PduR	PduR/PduRRoutingTables/PduRTpBufferTable	
BSW Parameter		**BSW Type**
PduRMaxTpBufferNumber		EcucIntegerParamDef
BSW Description		
maximum number of TP buffers.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**

BSW Module	**BSW Context**	
PduR	PduR/PduRRoutingTables/PduRTpBufferTable	
BSW Parameter		**BSW Type**
PduRTpBuffer		EcucParamConfContainerDef
BSW Description		
Specifies a buffer used for gatwaying through TP.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
PduR	PduR/PduRRoutingTables/PduRTpBufferTable/PduRTpBuffer	
BSW Parameter		**BSW Type**

PduRTpBufferLength		EcucIntegerParamDef	
BSW Description			
Length of the TP buffer in number of bytes			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**

BSW Module	**BSW Context**		
PduR	PduR/PduRRoutingTables		
BSW Parameter		**BSW Type**	
PduRTxBufferTable		EcucParamConfContainerDef	
BSW Description			
This container will specify the needed buffers for gatewaying using communication interface. It not defined per routing path to allow reusage of buffers.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	**BSW Context**		
PduR	PduR/PduRRoutingTables/PduRTxBufferTable		
BSW Parameter		**BSW Type**	
PduRMaxTxBufferNumber		EcucIntegerParamDef	
BSW Description			
maximum number of Tx buffers.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**

BSW Module	**BSW Context**		
PduR	PduR/PduRRoutingTables/PduRTxBufferTable		
BSW Parameter		**BSW Type**	
PduRTxBuffer		EcucParamConfContainerDef	
BSW Description			
Specifies a buffer used for gatwaying through communication interface.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	BSW Context		
PduR	PduR/PduRRoutingTables/PduRTxBufferTable/PduRTxBuffer		
BSW Parameter		**BSW Type**	
PduRPduMaxLength		EcucIntegerParamDef	
BSW Description			
Length of the Tx buffer in number of bytes.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**

BSW Module	BSW Context		
PduR	PduR/PduRRoutingTables/PduRTxBufferTable/PduRTxBuffer		
BSW Parameter		**BSW Type**	
PduRTxBufferDepth		EcucIntegerParamDef	
BSW Description			
Number of Pdus that can be stored in the buffer. If value is 1 then the buffer semantic is "last is best". If the value is greater then 1 then the buffer semnatic is a FiFo.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**

D.20 IPdu Multiplexer Mapping

BSW Module	BSW Context		
IpduM	IpduM		
BSW Parameter		**BSW Type**	
IpduMConfig		EcucParamConfContainerDef	
BSW Description			
This container contains the sub containers of the IpduM module. The IpduMTxPathway subcontainer includes information about sent I-PDUs. The IpduMRxPathway includes information about received I-PDUs. This container is a MultipleConfigurationContainer, i.e. this container and its sub-containers exist once per configuration set.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	BSW Context		
IpduM	IpduM/IpduMConfig		
BSW Parameter		**BSW Type**	

IpduMRxPathway			EcucParamConfContainerDef
BSW Description			
Contains the configuration parameters received I-PDUs by the IpduM module.			
M2 Template	**M2 Description**		
SystemTemplate	A MultiplexedPdu (i.e. NOT a COM I-PDU) contains a DynamicPart, an optional StaticPart and a selectorField. In case of multiplexing this IPdu is routed between the Pdu Multiplexer and the Interface Layer.		
M2 Parameter			
Fibex::FibexCore::CoreCommunication::MultiplexedIPdu			
Mapping Rule			**Mapping Type**
Create container for each received multiplexed Ipdu (IPduTriggering that references the MultiplexedIPdu contains a reference to an "In" Pdu Port.			full

BSW Module	**BSW Context**		
IpduM	IpduM/IpduMConfig/IpduMRxPathway		
BSW Parameter		**BSW Type**	
IpduMRxIndication		EcucParamConfContainerDef	
BSW Description			
Contains the configuration for incoming RxIndication calls.			
M2 Template	**M2 Description**		
SystemTemplate	A MultiplexedPdu (i.e. NOT a COM I-PDU) contains a DynamicPart, an optional StaticPart and a selectorField. In case of multiplexing this IPdu is routed between the Pdu Multiplexer and the Interface Layer.		
M2 Parameter			
Fibex::FibexCore::CoreCommunication::MultiplexedIPdu			
Mapping Rule			**Mapping Type**
Create container for each received multiplexed Ipdu (IPduTriggering that references the MultiplexedIPdu contains a reference to an "In" Pdu Port			full

BSW Module	**BSW Context**		
IpduM	IpduM/IpduMConfig/IpduMRxPathway/IpduMRxIndication		
BSW Parameter		**BSW Type**	
IpduMByteOrder		EcucEnumerationParamDef	
BSW Description			
This parameter defines the ByteOrder for all IpduMSegments (static and dynamic part) and for the selectorField within the MultiplexedPdu. The absolute position of a segment in the MultiplexedIPdu is determined by the definition of the ByteOrder parameter: If BIG_ENDIAN is specified, the SegmentPosition indicates the bit position of the most significant bit in an IPDU. If LITTLE_ENDIAN is specified, the SegmentPosition indicates the bit position of the least significant bit in an IPDU.			
M2 Template	**M2 Description**		
System Template	This attribute defines the order of the bytes of the segment and the packing into the MultiplexedIPdu.		
M2 Parameter			
FibexCore::DynamicPart.segmentByteOrder and FibexCore::StaticPart.segmentByteOrder and FibexCore::MultiplexedIPdu.selectorFieldByteOrder			
Mapping Rule			**Mapping Type**
A mix between Little Endian and Big Endian within a MultiplexedIPdu is not allowed.			full

BSW Module	BSW Context	
IpduM	IpduM/IpduMConfig/IpduMRxPathway/IpduMRxIndication	
BSW Parameter		**BSW Type**
IpduMRxDynamicPart		EcucParamConfContainerDef
BSW Description		
This container contains the configuration for the dynamic part of incoming RxIndication calls. When an incoming received I-PDU's selector field matches the IpduM_Selector_Value, the new outgoing I-PDU for the dynamic part is constructed as defined by the segments of this container and sent out with the I-PDU ID referenced by IpduMOutgoingDynamicPduRef.		
M2 Template	**M2 Description**	
SystemTemplate	One of the Com IPdu alternatives that are transmitted in the Dynamic Part of the MultiplexedIPdu. The selectorFieldCode specifies which Com IPdu is contained in the DynamicPart within a certain transmission of a multiplexed PDU.	
M2 Parameter		
Fibex::FibexCore::CoreCommunication::DynamicPartAlternative		
Mapping Rule		**Mapping Type**
Create container for each DynamicPartAlternative of the MultiplexedIPdu.		full

BSW Module	BSW Context	
IpduM	IpduM/IpduMConfig/IpduMRxPathway/IpduMRxIndication/IpduMRxDynamicPart	
BSW Parameter		**BSW Type**
IpduMOutgoingDynamicPduRef		EcucReferenceDef
BSW Description		
When the new I-PDU is sent out it is sent with this I-PDU ID. Reference to the sent PDU representation in the ECU Configuration Description exchange file.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
IpduM	IpduM/IpduMConfig/IpduMRxPathway/IpduMRxIndication/IpduMRxDynamicPart	
BSW Parameter		**BSW Type**
IpduMRxSelectorValue		EcucIntegerParamDef
BSW Description		
This is the selector value that this container refers to.		
M2 Template	**M2 Description**	
SystemTemplate	The selector field is part of a multiplexed IPdu. It consists of contiguous bits. The value of the selector field selects the layout of the multiplexed part of the IPdu.	
M2 Parameter		
Fibex::FibexCore::CoreCommunication::DynamicPartAlternative.selectorFieldCode		
Mapping Rule		**Mapping Type**
1:1 mapping		full

BSW Module	BSW Context

IpduM	IpduM/IpduMConfig/IpduMRxPathway/IpduMRxIndication/IpduMRxDynamic Part		
BSW Parameter		**BSW Type**	
IpduMSegment		EcucParamConfContainerDef	
BSW Description			
This contains the location and the length of a segment. A segment must fit inside the I-PDU. The segment in the source I-PDU that is located at the IpduMSegmentPosition is copied to the same position in the destination I-PDU.			
M2 Template	**M2 Description**		
SystemTemplate	The StaticPart and the DynamicPart can be separated in multiple segments within the multiplexed PDU.		
M2 Parameter			
Fibex::FibexCore::CoreCommunication::SegmentPosition			
Mapping Rule			**Mapping Type**
Source bit fields and the destination bit position can be derived from the segmentPosition.			full

BSW Module	**BSW Context**		
IpduM	IpduM/IpduMConfig/IpduMRxPathway/IpduMRxIndication/IpduMRxDynamic Part/IpduMSegment		
BSW Parameter		**BSW Type**	
IpduMSegmentLength		EcucIntegerParamDef	
BSW Description			
Length of the segment in bits.			
M2 Template	**M2 Description**		
SystemTemplate	With the attributes startPositionMultiplexedIPdu and startPositionInSignalIPdu a segment in the source can be copied to the segment in the destination.		
M2 Parameter			
Fibex::FibexCore::CoreCommunication::SegmentPosition.startPositionInMultiplexedIPdu or Fibex::FibexCore::CoreCommunication::SegmentPosition.startPositionInSignalIPdu			
Mapping Rule			**Mapping Type**
For TXRequest use startPositionInMultiplexedIPdu attribute. For RXIndication use startPositionInSignalPdu attribute.			full

BSW Module	**BSW Context**		
IpduM	IpduM/IpduMConfig/IpduMRxPathway/IpduMRxIndication/IpduMRxDynamic Part/IpduMSegment		
BSW Parameter		**BSW Type**	
IpduMSegmentPosition		EcucIntegerParamDef	
BSW Description			
Segments bit position in the multiplexed Pdu.			
M2 Template	**M2 Description**		
SystemTemplate	With the attributes startPositionMultiplexedIPdu and startPositionInSignalIPdu a segment in the source can be copied to the segment in the destination.		
M2 Parameter			
Fibex::FibexCore::CoreCommunication::SegmentPosition.startPositionInMultiplexedIPdu or Fibex::FibexCore::CoreCommunication::SegmentPosition.startPositionInSignalIPdu			
Mapping Rule			**Mapping Type**
For TXRequest use startPositionInMultiplexedIPdu attribute. For RXIndication use startPositionInSignalPdu attribute.			full

BSW Module	BSW Context	
IpduM	IpduM/IpduMConfig/IpduMRxPathway/IpduMRxIndication	
BSW Parameter		**BSW Type**
IpduMRxHandleId		EcucIntegerParamDef
BSW Description		
This is the I-PDU ID of the incoming I-PDU. If an incoming RxIndication's I-PDU ID matches this value then it is unpacked according to the specification in this container.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
IpduM	IpduM/IpduMConfig/IpduMRxPathway/IpduMRxIndication	
BSW Parameter		**BSW Type**
IpduMRxIndicationPduRef		EcucReferenceDef
BSW Description		
Reference to the received Pdu representation in the ECU Configuration Description exchange file.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
IpduM	IpduM/IpduMConfig/IpduMRxPathway/IpduMRxIndication	
BSW Parameter		**BSW Type**
IpduMRxStaticPart		EcucParamConfContainerDef
BSW Description		
This container contains the configuration for the static part of incoming RxIndication calls. On reception, the new outgoing I-PDU for the static part is constructed as defined by the segments of this container and sent out with the I-PDU ID referenced by IpduMOutgoingStaticPduRef.		
M2 Template	**M2 Description**	
SystemTemplate	Some parts/signals of the I-PDU may be the same regardless of the selector field. Such a part is called static part. The static part is optional.	
M2 Parameter		
Fibex::FibexCore::CoreCommunication::StaticPart		
Mapping Rule		**Mapping Type**
Create container if StaticPart exists in the MultiplexedIPdu.		full

BSW Module	BSW Context	
IpduM	IpduM/IpduMConfig/IpduMRxPathway/IpduMRxIndication/IpduMRxStaticPart	
BSW Parameter		**BSW Type**
IpduMOutgoingStaticPduRef		EcucReferenceDef
BSW Description		
When the new I-PDU is sent out it is sent with this I-PDU ID. Reference to the sent Pdu representation in the ECU Configuration Description exchange file.		

M2 Template	M2 Description	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
IpduM	IpduM/IpduMConfig/IpduMRxPathway/IpduMRxIndication/IpduMRxStaticPart	
BSW Parameter		**BSW Type**
IpduMSegment		EcucParamConfContainerDef
BSW Description		
This contains the location and the length of a segment. A segment must fit inside the I-PDU. The segment in the source I-PDU that is located at the IpduMSegmentPosition is copied to the same position in the destination I-PDU.		
M2 Template	**M2 Description**	
SystemTemplate	The StaticPart and the DynamicPart can be separated in multiple segments within the multiplexed PDU.	
M2 Parameter		
Fibex::FibexCore::CoreCommunication::SegmentPosition		
Mapping Rule		**Mapping Type**
Source bit fields and the destination bit position can be derived from the segmentPosition.		full

BSW Module	BSW Context	
IpduM	IpduM/IpduMConfig/IpduMRxPathway/IpduMRxIndication/IpduMRxStaticPart/IpduMSegment	
BSW Parameter		**BSW Type**
IpduMSegmentLength		EcucIntegerParamDef
BSW Description		
Length of the segment in bits.		
M2 Template	**M2 Description**	
SystemTemplate	With the attributes startPositionMultiplexedIPdu and startPositionInSignalIPdu a segment in the source can be copied to the segment in the destination.	
M2 Parameter		
Fibex::FibexCore::CoreCommunication::SegmentPosition.startPositionInMultiplexedIPdu or Fibex::FibexCore::CoreCommunication::SegmentPosition.startPositionInSignalIPdu		
Mapping Rule		**Mapping Type**
For TXRequest use startPositionInMultiplexedIPdu attribute. For RXIndication use startPositionInSignalPdu attribute.		full

BSW Module	BSW Context	
IpduM	IpduM/IpduMConfig/IpduMRxPathway/IpduMRxIndication/IpduMRxStaticPart/IpduMSegment	
BSW Parameter		**BSW Type**
IpduMSegmentPosition		EcucIntegerParamDef
BSW Description		
Segments bit position in the multiplexed Pdu.		
M2 Template	**M2 Description**	
SystemTemplate	With the attributes startPositionMultiplexedIPdu and startPositionInSignalIPdu a segment in the source can be copied to the segment in the destination.	

M2 Parameter	
Fibex::FibexCore::CoreCommunication::SegmentPosition.startPositionInMultiplexedIPdu or Fibex::FibexCore::CoreCommunication::SegmentPosition.startPositionInSignalIPdu	
Mapping Rule	**Mapping Type**
For TXRequest use startPositionInMultiplexedIPdu attribute. For RXIndication use startPositionInSignalPdu attribute.	full

BSW Module	BSW Context	
IpduM	IpduM/IpduMConfig/IpduMRxPathway/IpduMRxIndication	
BSW Parameter		**BSW Type**
IpduMSelectorFieldPosition		EcucParamConfContainerDef
BSW Description		
This contains the location and the length of the selector field.		
M2 Template	**M2 Description**	
SystemTemplate	The StaticPart and the DynamicPart can be separated in multiple segments within the multiplexed PDU.	
M2 Parameter		
Fibex::FibexCore::CoreCommunication::SegmentPosition		
Mapping Rule		**Mapping Type**
Can be derived from the segmentPosition.		full

BSW Module	BSW Context	
IpduM	IpduM/IpduMConfig/IpduMRxPathway/IpduMRxIndication/IpduMSelectorFieldPosition	
BSW Parameter		**BSW Type**
IpduMSelectorFieldLength		EcucIntegerParamDef
BSW Description		
Length of the selector field in bits.		
M2 Template	**M2 Description**	
SystemTemplate	With the attributes startPositionMultiplexedIPdu and startPositionInSignalIPdu a segment in the source can be copied to the segment in the destination.	
M2 Parameter		
Fibex::FibexCore::CoreCommunication::SegmentPosition.startPositionInMultiplexedIPdu or Fibex::FibexCore::CoreCommunication::SegmentPosition.startPositionInSignalIPdu		
Mapping Rule		**Mapping Type**
For TXRequest use startPositionInMultiplexedIPdu attribute. For RXIndication use startPositionInSignalPdu attribute.		full

BSW Module	BSW Context	
IpduM	IpduM/IpduMConfig/IpduMRxPathway/IpduMRxIndication/IpduMSelectorFieldPosition	
BSW Parameter		**BSW Type**
IpduMSelectorFieldPosition		EcucIntegerParamDef
BSW Description		
Selector field bit position in the multiplexed Pdu.		
M2 Template	**M2 Description**	
SystemTemplate	With the attributes startPositionMultiplexedIPdu and startPositionInSignalIPdu a segment in the source can be copied to the segment in the destination.	
M2 Parameter		
Fibex::FibexCore::CoreCommunication::SegmentPosition.startPositionInMultiplexedIPdu or Fibex::FibexCore::CoreCommunication::SegmentPosition.startPositionInSignalIPdu		

Mapping Rule	Mapping Type
For TXRequest use startPositionInMultiplexedIPdu attribute. For RXIndication use startPositionInSignalPdu attribute.	full

BSW Module	BSW Context	
IpduM	IpduM/IpduMConfig	
BSW Parameter		**BSW Type**
IpduMTxPathway		EcucParamConfContainerDef
BSW Description		
Contains the configuration parameters transmitted I-PDUs by the IpduM module.		
M2 Template	**M2 Description**	
SystemTemplate	A MultiplexedPdu (i.e. NOT a COM I-PDU) contains a DynamicPart, an optional StaticPart and a selectorField. In case of multiplexing this IPdu is routed between the Pdu Multiplexer and the Interface Layer.	
M2 Parameter		
Fibex::FibexCore::CoreCommunication::MultiplexedIPdu		
Mapping Rule		**Mapping Type**
Create container for each transmitted multiplexed Ipdu (IPduTriggering that references the MultiplexedIPdu contains a reference to an "Out" Pdu Port.		full

BSW Module	BSW Context	
IpduM	IpduM/IpduMConfig/IpduMTxPathway	
BSW Parameter		**BSW Type**
IpduMTxRequest		EcucParamConfContainerDef
BSW Description		
This is used to specify the configuration for Transmit requests. There will one instance of this container for each I-PDU that can be requested for transmission (the outgoing I-PDUs) by the IpduM.		
M2 Template	**M2 Description**	
SystemTemplate	A MultiplexedPdu (i.e. NOT a COM I-PDU) contains a DynamicPart, an optional StaticPart and a selectorField. In case of multiplexing this IPdu is routed between the Pdu Multiplexer and the Interface Layer.	
M2 Parameter		
Fibex::FibexCore::CoreCommunication::MultiplexedIPdu		
Mapping Rule		**Mapping Type**
Create container for each transmitted multiplexed Ipdu		full

BSW Module	BSW Context	
IpduM	IpduM/IpduMConfig/IpduMTxPathway/IpduMTxRequest	
BSW Parameter		**BSW Type**
IpduMByteOrder		EcucEnumerationParamDef
BSW Description		
This parameter defines the ByteOrder for all IpduMSegments (static and dynamic part) and for the selectorField within the MultiplexedPdu. The absolute position of a segment in the MultiplexedIPdu is determined by the definition of the ByteOrder parameter: If BIG_ENDIAN is specified, the SegmentPosition indicates the bit position of the most significant bit in an IPDU. If LITTLE_ENDIAN is specified, the SegmentPosition indicates the bit position of the least significant bit in an IPDU.		

M2 Template	**M2 Description**
System Template	This attribute defines the order of the bytes of the segment and the packing into the MultiplexedIPdu.
M2 Parameter	
FibexCore::DynamicPart.segmentByteOrder and FibexCore::StaticPart.segmentByteOrder and FibexCore::MultiplexedIPdu.selectorFieldByteOrder	
Mapping Rule	**Mapping Type**
A mix between Little Endian and Big Endian within a MultiplexedIPdu is not allowed.	full

BSW Module	**BSW Context**
IpduM	IpduM/IpduMConfig/IpduMTxPathway/IpduMTxRequest
BSW Parameter	**BSW Type**
IpduMIPduUnusedAreasDefault	EcucIntegerParamDef
BSW Description	
IpduM module fills not used areas of an I-PDU with this bit-pattern If this attribute is omitted the IpduM module does not fill the I-PDU.	
M2 Template	**M2 Description**
SystemTemplate	AUTOSAR COM fills not used areas of an IPDU with this bit-pattern. This attribute is mandatory to avoid undefined behavior. This byte-pattern will be repeated throughout the IPDU.
M2 Parameter	
Fibex::FibexCore::CoreCommunication::MultiplexedIPdu.unusedBitPattern	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	**BSW Context**
IpduM	IpduM/IpduMConfig/IpduMTxPathway/IpduMTxRequest
BSW Parameter	**BSW Type**
IpduMInitialDynamicPart	EcucReferenceDef
BSW Description	
Reference to the dynamic part that shall be used to initialize this multiplexed TX-I-PDU.	
M2 Template	**M2 Description**
SystemTemplate	Dynamic part that shall be used to initialize this multiplexed IPdu.
M2 Parameter	
Fibex::FibexCore::CoreCommunication::DynamicPartAlternative.initialDynamicPart	
Mapping Rule	**Mapping Type**
If the attribute initialDynamicPart is set to true then create this reference.	full

BSW Module	**BSW Context**
IpduM	IpduM/IpduMConfig/IpduMTxPathway/IpduMTxRequest
BSW Parameter	**BSW Type**
IpduMOutgoingPduRef	EcucReferenceDef
BSW Description	
Reference to the PDU defining the outgoing I-PDU. When the outgoing I-PDU is sent this is the I-PDU ID to give it. It is the IpduM I-PDU ID of the assembled I-PDU.	
M2 Template	**M2 Description**
M2 Parameter	

Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**	
IpduM	IpduM/IpduMConfig/IpduMTxPathway/IpduMTxRequest	
BSW Parameter		**BSW Type**
IpduMSelectorFieldPosition		EcucParamConfContainerDef
BSW Description		
This contains the location and the length of the selector field.		
M2 Template	**M2 Description**	
SystemTemplate	The StaticPart and the DynamicPart can be separated in multiple segments within the multiplexed PDU.	
M2 Parameter		
Fibex::FibexCore::CoreCommunication::SegmentPosition		
Mapping Rule		**Mapping Type**
Can be derived from the segmentPosition.		full

BSW Module	**BSW Context**	
IpduM	IpduM/IpduMConfig/IpduMTxPathway/IpduMTxRequest/IpduMSelectorField Position	
BSW Parameter		**BSW Type**
IpduMSelectorFieldLength		EcucIntegerParamDef
BSW Description		
Length of the selector field in bits.		
M2 Template	**M2 Description**	
SystemTemplate	With the attributes startPositionMultiplexedIPdu and startPositionInSignalIPdu a segment in the source can be copied to the segment in the destination.	
M2 Parameter		
Fibex::FibexCore::CoreCommunication::SegmentPosition.startPositionInMultiplexedIPdu or Fibex::FibexCore::CoreCommunication::SegmentPosition.startPositionInSignalIPdu		
Mapping Rule		**Mapping Type**
For TXRequest use startPositionInMultiplexedIPdu attribute. For RXIndication use startPositionInSignalPdu attribute.		full

BSW Module	**BSW Context**	
IpduM	IpduM/IpduMConfig/IpduMTxPathway/IpduMTxRequest/IpduMSelectorField Position	
BSW Parameter		**BSW Type**
IpduMSelectorFieldPosition		EcucIntegerParamDef
BSW Description		
Selector field bit position in the multiplexed Pdu.		
M2 Template	**M2 Description**	
SystemTemplate	With the attributes startPositionMultiplexedIPdu and startPositionInSignalIPdu a segment in the source can be copied to the segment in the destination.	
M2 Parameter		
Fibex::FibexCore::CoreCommunication::SegmentPosition.startPositionInMultiplexedIPdu or Fibex::FibexCore::CoreCommunication::SegmentPosition.startPositionInSignalIPdu		
Mapping Rule		**Mapping Type**
For TXRequest use startPositionInMultiplexedIPdu attribute. For RXIndication use startPositionInSignalPdu attribute.		full

<table>
<tr><th>BSW Module</th><th colspan="2">BSW Context</th></tr>
<tr><td>IpduM</td><td colspan="2">IpduM/IpduMConfig/IpduMTxPathway/IpduMTxRequest</td></tr>
<tr><th colspan="2">BSW Parameter</th><th>BSW Type</th></tr>
<tr><td colspan="2">IpduMTxConfirmationPduId</td><td>EcucIntegerParamDef</td></tr>
<tr><th colspan="3">BSW Description</th></tr>
<tr><td colspan="3">The handle Id to be used by the PduR to confirm the transmission of this Pdu.

The existence of this parameter is essential for the PduR generation tool to actually find a symbolicNameValue for the OutgoingPdu.</td></tr>
<tr><th>M2 Template</th><th colspan="2">M2 Description</th></tr>
<tr><td></td><td colspan="2"></td></tr>
<tr><th colspan="3">M2 Parameter</th></tr>
<tr><td colspan="3"></td></tr>
<tr><th colspan="2">Mapping Rule</th><th>Mapping Type</th></tr>
<tr><td colspan="2"></td><td></td></tr>
</table>

<table>
<tr><th>BSW Module</th><th colspan="2">BSW Context</th></tr>
<tr><td>IpduM</td><td colspan="2">IpduM/IpduMConfig/IpduMTxPathway/IpduMTxRequest</td></tr>
<tr><th colspan="2">BSW Parameter</th><th>BSW Type</th></tr>
<tr><td colspan="2">IpduMTxConfirmationTimeout</td><td>EcucFloatParamDef</td></tr>
<tr><th colspan="3">BSW Description</th></tr>
<tr><td colspan="3">This timeout (in seconds) defines the timeout period for monitoring the reception of the TxConfirmation.
It is not used when an I-PDU is requested using the trigger transmit API.</td></tr>
<tr><th>M2 Template</th><th colspan="2">M2 Description</th></tr>
<tr><td></td><td colspan="2"></td></tr>
<tr><th colspan="3">M2 Parameter</th></tr>
<tr><td colspan="3"></td></tr>
<tr><th colspan="2">Mapping Rule</th><th>Mapping Type</th></tr>
<tr><td colspan="2"></td><td></td></tr>
</table>

<table>
<tr><th>BSW Module</th><th colspan="2">BSW Context</th></tr>
<tr><td>IpduM</td><td colspan="2">IpduM/IpduMConfig/IpduMTxPathway/IpduMTxRequest</td></tr>
<tr><th colspan="2">BSW Parameter</th><th>BSW Type</th></tr>
<tr><td colspan="2">IpduMTxDynamicPart</td><td>EcucParamConfContainerDef</td></tr>
<tr><th colspan="3">BSW Description</th></tr>
<tr><td colspan="3">Configuration parameters for an instance of a TxRequest call into the IpduM. When a Tx Request with the IpduMTxDynamicHandleId is received by the IpduM, all segments as defined by this container are copied from the incoming I-PDU into the outgoing I-PDU buffer and then the send mode honoured. This container is used by the dynamic part of a TxRequest configuration. Therefore, for each outgoing I-PDU there will be one instance of this container for the dynamic part.</td></tr>
<tr><th>M2 Template</th><th colspan="2">M2 Description</th></tr>
<tr><td>SystemTemplate</td><td colspan="2">One of the Com IPdu alternatives that are transmitted in the Dynamic Part of the MultiplexedIPdu. The selectorFieldCode specifies which Com IPdu is contained in the DynamicPart within a certain transmission of a multiplexed PDU.</td></tr>
<tr><th colspan="3">M2 Parameter</th></tr>
<tr><td colspan="3">Fibex::FibexCore::CoreCommunication::DynamicPartAlternative</td></tr>
<tr><th colspan="2">Mapping Rule</th><th>Mapping Type</th></tr>
<tr><td colspan="2">Create container for each DynamicPartAlternative of the MultiplexedIPdu.</td><td>full</td></tr>
</table>

BSW Module	BSW Context

IpduM	IpduM/IpduMConfig/IpduMTxPathway/IpduMTxRequest/IpduMTxDynamicPart		
BSW Parameter		**BSW Type**	
IpduMJitUpdate		EcucBooleanParamDef	
BSW Description			
If configured to true fetch the data of this part Just-In-Time via the triggerTransmit API of the PduR.			
M2 Template	**M2 Description**		
M2 Parameter			
Mapping Rule			**Mapping Type**
			local

BSW Module	**BSW Context**		
IpduM	IpduM/IpduMConfig/IpduMTxPathway/IpduMTxRequest/IpduMTxDynamicPart		
BSW Parameter		**BSW Type**	
IpduMSegment		EcucParamConfContainerDef	
BSW Description			
This contains the location and the length of a segment. A segment must fit inside the I-PDU. The segment in the source I-PDU that is located at the IpduMSegmentPosition is copied to the same position in the destination I-PDU.			
M2 Template	**M2 Description**		
SystemTemplate	The StaticPart and the DynamicPart can be separated in multiple segments within the multiplexed PDU.		
M2 Parameter			
Fibex::FibexCore::CoreCommunication::SegmentPosition			
Mapping Rule			**Mapping Type**
Source bit fields and the destination bit position can be derived from the segmentPosition.			full

BSW Module	**BSW Context**		
IpduM	IpduM/IpduMConfig/IpduMTxPathway/IpduMTxRequest/IpduMTxDynamicPart/IpduMSegment		
BSW Parameter		**BSW Type**	
IpduMSegmentLength		EcucIntegerParamDef	
BSW Description			
Length of the segment in bits.			
M2 Template	**M2 Description**		
SystemTemplate	With the attributes startPositionMultiplexedIPdu and startPositionInSignalIPdu a segment in the source can be copied to the segment in the destination.		
M2 Parameter			
Fibex::FibexCore::CoreCommunication::SegmentPosition.startPositionInMultiplexedIPdu or Fibex::FibexCore::CoreCommunication::SegmentPosition.startPositionInSignalIPdu			
Mapping Rule			**Mapping Type**
For TXRequest use startPositionInMultiplexedIPdu attribute. For RXIndication use startPositionInSignalPdu attribute.			full

BSW Module	**BSW Context**	
IpduM	IpduM/IpduMConfig/IpduMTxPathway/IpduMTxRequest/IpduMTxDynamicPart/IpduMSegment	
BSW Parameter		**BSW Type**
IpduMSegmentPosition		EcucIntegerParamDef

BSW Description	
Segments bit position in the multiplexed Pdu.	
M2 Template	**M2 Description**
SystemTemplate	With the attributes startPositionMultiplexedIPdu and startPositionInSignalIPdu a segment in the source can be copied to the segment in the destination.
M2 Parameter	
Fibex::FibexCore::CoreCommunication::SegmentPosition.startPositionInMultiplexedIPdu or Fibex::FibexCore::CoreCommunication::SegmentPosition.startPositionInSignalIPdu	
Mapping Rule	**Mapping Type**
For TXRequest use startPositionInMultiplexedIPdu attribute. For RXIndication use startPositionInSignalPdu attribute.	full

BSW Module	BSW Context
IpduM	IpduM/IpduMConfig/IpduMTxPathway/IpduMTxRequest/IpduMTxDynamicPart
BSW Parameter	**BSW Type**
IpduMTxDynamicConfirmation	EcucBooleanParamDef
BSW Description	
A transmit request can be confirmed by the lower layer. If this parameter is set to true a confirmation of the I-PDU in COM representing the dynamic part is generated.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	BSW Context
IpduM	IpduM/IpduMConfig/IpduMTxPathway/IpduMTxRequest/IpduMTxDynamicPart
BSW Parameter	**BSW Type**
IpduMTxDynamicHandleId	EcucIntegerParamDef
BSW Description	
This is an incoming handle id. When the handle of an incoming Tx Request matches this, the bits fields (see IpduMSegment) are copied and the IpduMTxTriggerMode is honored.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	BSW Context
IpduM	IpduM/IpduMConfig/IpduMTxPathway/IpduMTxRequest/IpduMTxDynamicPart
BSW Parameter	**BSW Type**
IpduMTxDynamicPduRef	EcucReferenceDef
BSW Description	
Reference to the Pdu representation in the ECU Configuration Description exchange file to be transmitted.	
M2 Template	**M2 Description**
M2 Parameter	

Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**	
IpduM	IpduM/IpduMConfig/IpduMTxPathway/IpduMTxRequest	
BSW Parameter		**BSW Type**
IpduMTxStaticPart		EcucParamConfContainerDef
BSW Description		
Configuration parameters for an instance of a Tx_Request call into the IpduM. When a Tx Request with the IpduMTxStaticHandleId is received by the IpduM, all segments as defined by this container are copied from the incoming I-PDU into the outgoing I-PDU buffer and then the send mode honoured. This container is used for the static part of a TxRequest configuration. Therefore, for each outgoing I-PDU there will be one instance of this container for the static part if it exists.		
M2 Template	**M2 Description**	
SystemTemplate	Some parts/signals of the I-PDU may be the same regardless of the selector field. Such a part is called static part. The static part is optional.	
M2 Parameter		
Fibex::FibexCore::CoreCommunication::StaticPart		
Mapping Rule		**Mapping Type**
Create container if StaticPart exists in the MultiplexedIPdu.		full

BSW Module	**BSW Context**	
IpduM	IpduM/IpduMConfig/IpduMTxPathway/IpduMTxRequest/IpduMTxStaticPart	
BSW Parameter		**BSW Type**
IpduMJitUpdate		EcucBooleanParamDef
BSW Description		
If configured to true fetch the data of this part Just-In-Time via the triggerTransmit API of the PduR.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	**BSW Context**	
IpduM	IpduM/IpduMConfig/IpduMTxPathway/IpduMTxRequest/IpduMTxStaticPart	
BSW Parameter		**BSW Type**
IpduMSegment		EcucParamConfContainerDef
BSW Description		
This contains the location and the length of a segment. A segment must fit inside the I-PDU. The segment in the source I-PDU that is located at the IpduMSegmentPosition is copied to the same position in the destination I-PDU.		
M2 Template	**M2 Description**	
SystemTemplate	The StaticPart and the DynamicPart can be separated in multiple segments within the multiplexed PDU.	
M2 Parameter		
Fibex::FibexCore::CoreCommunication::SegmentPosition		
Mapping Rule		**Mapping Type**
Source bit fields and the destination bit position can be derived from the segmentPosition.		full

BSW Module	BSW Context	
IpduM	IpduM/IpduMConfig/IpduMTxPathway/IpduMTxRequest/IpduMTxStaticPart/IpduMSegment	
BSW Parameter		**BSW Type**
IpduMSegmentLength		EcucIntegerParamDef
BSW Description		
Length of the segment in bits.		
M2 Template	**M2 Description**	
SystemTemplate	With the attributes startPositionMultiplexedIPdu and startPositionInSignalIPdu a segment in the source can be copied to the segment in the destination.	
M2 Parameter		
Fibex::FibexCore::CoreCommunication::SegmentPosition.startPositionInMultiplexedIPdu or Fibex::FibexCore::CoreCommunication::SegmentPosition.startPositionInSignalIPdu		
Mapping Rule		**Mapping Type**
For TXRequest use startPositionInMultiplexedIPdu attribute. For RXIndication use startPositionInSignalPdu attribute.		full

BSW Module	BSW Context	
IpduM	IpduM/IpduMConfig/IpduMTxPathway/IpduMTxRequest/IpduMTxStaticPart/IpduMSegment	
BSW Parameter		**BSW Type**
IpduMSegmentPosition		EcucIntegerParamDef
BSW Description		
Segments bit position in the multiplexed Pdu.		
M2 Template	**M2 Description**	
SystemTemplate	With the attributes startPositionMultiplexedIPdu and startPositionInSignalIPdu a segment in the source can be copied to the segment in the destination.	
M2 Parameter		
Fibex::FibexCore::CoreCommunication::SegmentPosition.startPositionInMultiplexedIPdu or Fibex::FibexCore::CoreCommunication::SegmentPosition.startPositionInSignalIPdu		
Mapping Rule		**Mapping Type**
For TXRequest use startPositionInMultiplexedIPdu attribute. For RXIndication use startPositionInSignalPdu attribute.		full

BSW Module	BSW Context	
IpduM	IpduM/IpduMConfig/IpduMTxPathway/IpduMTxRequest/IpduMTxStaticPart	
BSW Parameter		**BSW Type**
IpduMTxStaticConfirmation		EcucBooleanParamDef
BSW Description		
A transmit request can be confirmed by the lower layer. If this parameter is set to true a confirmation of the I-PDU in COM representing the static part is generated.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
IpduM	IpduM/IpduMConfig/IpduMTxPathway/IpduMTxRequest/IpduMTxStaticPart	
BSW Parameter		**BSW Type**

IpduMTxStaticHandleId	EcucIntegerParamDef
BSW Description	
This is an incoming handle id. When the handle of an incoming Tx Request matches this, the segments are copied (IPduMSegment) and the IpduMTxTriggerMode is honored.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**
IpduM	IpduM/IpduMConfig/IpduMTxPathway/IpduMTxRequest/IpduMTxStaticPart
BSW Parameter	**BSW Type**
IpduMTxStaticPduRef	EcucReferenceDef
BSW Description	
Reference to the Pdu representation in the ECU Configuration Description exchange file to be transmitted.	
M2 Template	**M2 Description**
M2 Parameter	
Mapping Rule	**Mapping Type**
	local

BSW Module	**BSW Context**
IpduM	IpduM/IpduMConfig/IpduMTxPathway/IpduMTxRequest
BSW Parameter	**BSW Type**
IpduMTxTriggerMode	EcucEnumerationParamDef
BSW Description	
Selects whether to send the multiplexed I-PDU immediately or at some later date.	
M2 Template	**M2 Description**
SystemTemplate	IPduM can be configured to send a transmission request for the new multiplexed I-PDU to the PDU-Router because of the trigger conditions/ modes that are described in the TriggerMode enumeration.
M2 Parameter	
Fibex::FibexCore::CoreCommunication::MultiplexedIPdu.triggerMode	
Mapping Rule	**Mapping Type**
1:1 mapping	full

BSW Module	**BSW Context**
IpduM	IpduM
BSW Parameter	**BSW Type**
IpduMGeneral	EcucParamConfContainerDef
BSW Description	
Contains the general configuration parameters of IpduM.	
M2 Template	**M2 Description**
M2 Parameter	

Mapping Rule	Mapping Type
	local

BSW Module	BSW Context	
IpduM	IpduM/IpduMGeneral	
BSW Parameter		**BSW Type**
IpduMConfigurationTimeBase		EcucFloatParamDef
BSW Description		
The cycle time with which IpduM_MainFunction should be invoked (in seconds).		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
IpduM	IpduM/IpduMGeneral	
BSW Parameter		**BSW Type**
IpduMDevErrorDetect		EcucBooleanParamDef
BSW Description		
Active/Deactivate the detection of development errors, for production code this parameter has to be False. True: error detection activated False: error detection deactivated		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

BSW Module	BSW Context	
IpduM	IpduM/IpduMGeneral	
BSW Parameter		**BSW Type**
IpduMStaticPartExists		EcucBooleanParamDef
BSW Description		
This is to allow optimizations in the case the IpduM will never be used with a static part. Note that this is a pre-compile option. If this is set to False then it will not be possible to add static parts after compilation. True: A static part may exist. False: A static part will never exist.		
M2 Template	**M2 Description**	
M2 Parameter		
Mapping Rule		**Mapping Type**
		local

<table>
<tr><th>BSW Module</th><th colspan="3">BSW Context</th></tr>
<tr><td>IpduM</td><td colspan="3">IpduM/IpduMGeneral</td></tr>
<tr><th colspan="2">BSW Parameter</th><th colspan="2">BSW Type</th></tr>
<tr><td colspan="2">IpduMVersionInfoApi</td><td colspan="2">EcucBooleanParamDef</td></tr>
<tr><th colspan="4">BSW Description</th></tr>
<tr><td colspan="4">Active/Deactivate the version information API.

true: version information activated
false: version information deactivated</td></tr>
<tr><th>M2 Template</th><th colspan="3">M2 Description</th></tr>
<tr><td></td><td colspan="3"></td></tr>
<tr><th colspan="4">M2 Parameter</th></tr>
<tr><td colspan="4"></td></tr>
<tr><th colspan="3">Mapping Rule</th><th>Mapping Type</th></tr>
<tr><td colspan="3"></td><td>local</td></tr>
</table>

<table>
<tr><th>BSW Module</th><th colspan="3">BSW Context</th></tr>
<tr><td>IpduM</td><td colspan="3">IpduM</td></tr>
<tr><th colspan="2">BSW Parameter</th><th colspan="2">BSW Type</th></tr>
<tr><td colspan="2">IpduMPublishedInformation</td><td colspan="2">EcucParamConfContainerDef</td></tr>
<tr><th colspan="4">BSW Description</th></tr>
<tr><td colspan="4">Additional published parameters not covered by CommonPublishedInformation container. Note that these parameters do not have any configuration class setting, since they are published information.</td></tr>
<tr><th>M2 Template</th><th colspan="3">M2 Description</th></tr>
<tr><td></td><td colspan="3"></td></tr>
<tr><th colspan="4">M2 Parameter</th></tr>
<tr><td colspan="4"></td></tr>
<tr><th colspan="3">Mapping Rule</th><th>Mapping Type</th></tr>
<tr><td colspan="3"></td><td></td></tr>
</table>

<table>
<tr><th>BSW Module</th><th colspan="3">BSW Context</th></tr>
<tr><td>IpduM</td><td colspan="3">IpduM/IpduMPublishedInformation</td></tr>
<tr><th colspan="2">BSW Parameter</th><th colspan="2">BSW Type</th></tr>
<tr><td colspan="2">IpduMRxDirectComInvocation</td><td colspan="2">EcucBooleanParamDef</td></tr>
<tr><th colspan="4">BSW Description</th></tr>
<tr><td colspan="4">If set to TRUE the COM invocation optimization as defined in IPDUM140 is implemented.</td></tr>
<tr><th>M2 Template</th><th colspan="3">M2 Description</th></tr>
<tr><td></td><td colspan="3"></td></tr>
<tr><th colspan="4">M2 Parameter</th></tr>
<tr><td colspan="4"></td></tr>
<tr><th colspan="3">Mapping Rule</th><th>Mapping Type</th></tr>
<tr><td colspan="3"></td><td></td></tr>
</table>

E Renamed Meta-Model Elements

E.1 Introduction

In the course of preparing AUTOSAR Release 4.0 some of the existing meta-model elements have been renamed for a better clarity and consistency with respect to other meta-mode elements. This chapter provides an overview of the changed meta-model elements in order to allow readers with a background in former specifications to understand changes made by mere renaming.

E.2 Renamed Meta-Model Elements

Old Name	*New Name*
AbsolutelyScheduledTiming	FlexRayAbsolutelyScheduledTiming
IPduGroup	ISignalIPduGroup
SignalPort	ISignalPort
SignalComposition	RootSwCompositionPrototype
IPduTriggering	PduTriggering

Table E.1: Renamed meta-model elements

F Constraint History

F.1 Constraint History of this Document according to AUTOSAR R4.0.1

F.1.1 Changed Constraints in R4.0.1

N/A

F.1.2 Added Constraints in R4.0.1

Number	Heading
[constr_3000]	valid SenderRecCompositeTypeMappings
[constr_3001]	valid ClientServerToSignalGroupMappings
[constr_3002]	valid SwcToImplMapping
[constr_3003]	Number of CAN channels
[constr_3004]	Clustering and separation must be exclusive
[constr_3005]	valid EcuResourceEstimation
[constr_3006]	valid EcuMapping
[constr_3007]	SelectorFieldCodes for dynamic part alternatives
[constr_3008]	EcuInstance subelements
[constr_3009]	Overlapping of ISignals is prohibited
[constr_3010]	ISignalIPdu shall not be exceeded
[constr_3011]	Overlapping of updateIndicationBits for ISignals is prohibited
[constr_3012]	Overlapping of Pdus is prohibited
[constr_3013]	Frame length shall not be exceeded
[constr_3014]	Overlapping of updateIndicationBits for Pdus is prohibited
[constr_3015]	Number of LIN channels
[constr_3016]	Number of Ethernet channels
[constr_3017]	Length of multiplexed Pdu shall not be exceeded
[constr_3018]	Number of FlexRay channels

Table F.1: Added Constraints in R4.0.1

F.1.3 Deleted Constraints in R4.0.1

N/A

F.2 Constraint History of this Document according to AUTOSAR R4.0.2

F.2.1 Changed Constraints in R4.0.2

N/A

F.2.2 Added Constraints in R4.0.2

Number	Heading
[constr_3019]	In the flat ECU extract each required interface must be satisfied by connected provided interfaces

Table F.2: Added Constraints in R4.0.2

F.2.3 Deleted Constraints in R4.0.2

N/A

F.3 Constraint and Specification Item History of this document according to AUTOSAR R4.0.3

F.3.1 Changed Constraints in R4.0.3

N/A

F.3.2 Changed Specification Items in R4.0.3

N/A

F.3.3 Added Constraints in R4.0.3

Number	Heading
[constr_3020]	CommmunicationDirection of containedIPduGroups
[constr_3021]	Mapping of `SensorActuatorSwComponents` to SensorActuator `HwElements`
[constr_3024]	Usage of `triggeredWithoutRepetition` and `triggeredOnChangeWithoutRepetition` is not allowed for signal groups and group signals.
[constr_3025]	Usage of NPdus in TpConnections
[constr_3026]	valid `EmptySignalMappings`

Table F.3: Added Constraints in R4.0.3

F.3.4 Added Specification Items in R4.0.3

Number	Heading
[TPS_SYST_1000]	`FlatInstanceDescriptor` roles

Table F.4: Added Specification Items in 4.0.3

F.3.5 Deleted Constraints in R4.0.3

N/A

F.3.6 Deleted Specification Items in R4.0.3

N/A

— AUTOSAR CONFIDENTIAL —